FORTIFIED END-TO-END LOCATION PRIVACY AND ANONYMITY IN WIRELESS SENSOR NETWORKS: A MODULAR APPROACH

Abdel-shakour A. Abuzneid

Under the Supervision of Dr. Tarek M. Sobh

DISSERTATION

SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENTS

FOR THE DEGREE OF DOCTOR OF PHILOSOPHY IN COMPUTER SCIENCE

AND ENGINEERING

THE SCHOOL OF ENGINEERING

UNIVERSITY OF BRIDGEPORT

CONNECTICUT

March 2015

FORTIFIED END-TO-END LOCATION PRIVACY AND ANONYMITY IN WIRELESS SENSOR NETWORKS: A MODULAR APPROACH

Approvals

Committee Members

Name	Signature	Date
Dr. Tarek M. Sobh	T. Sobh	3-20-2015
Dr. Miad Faezipour	Miad Faezipour	03/13/15
Dr. Ausif Mahmood	Ausif Mahmood	3-13-2015
Dr. Khaled M. Elleithy	K. Elleithy	3/13/15
Dr. John James	John James	

Ph.D. Program Coordinator

Dr. Khaled M. Elleithy	K. Elleithy	3/20/15

Chairperson, Computer Science and Engineering Department

Dr. Ausif Mahmood	Ausif Mahmood	3-13-2015

Dean, School of Engineering

Dr. Tarek M. Sobh	T. Sobh	3-20-2015

FORTIFIED END-TO-END LOCATION PRIVACY AND ANONYMITY IN WIRELESS SENSOR NETWORKS: A MODULAR APPROACH

FORTIFIED END-TO-END LOCATION PRIVACY AND ANONYMITY IN WIRELESS SENSOR NETWORKS: A MODULAR APPROACH ABSTRACT

ABSTRACT

Wireless sensor network (WSN) consists of many hosts called sensors. These sensors can sense a phenomenon (motion, temperature, humidity, average, max, min, etc.) and represent what they sense in a form of data. There are many applications for WSNs; including object tracking and monitoring where in most of the cases these objects need protection. In these applications, data *privacy* itself might not be as important as the privacy of source location. In addition to the source location privacy, sink location privacy should also be provided. Providing an efficient end-to-end privacy solution would be a challenging task to achieve due to the open nature of the WSN. The key schemes needed for end-to-end location privacy are anonymity, observability, capture likelihood, and safety period. We extend this work to allow for countermeasures against multi-local and global adversaries. We present a network model that is protected against a sophisticated threat model: passive /active and local/multi-local/global attacks. This work provides a solution for end-to-end anonymity and location privacy as well. We will introduce a framework called fortified anonymous communication (FAC) protocol for WSN.

ACKNOWLEDGEMENTS

My thanks are wholly devoted to God who has helped me all the way to complete this work successfully. I owe a debt of gratitude to my family, my father, my mother, my lovely wife and my three precious children for understanding and encouragement.

I am honored that my work has been supervised by Dr. Tarek Sobh and Dr. Miad Faezipour. I truly appreciate all they have provide to me. Many thanks to the committee members, Dr. Ausif Mahmoud, Dr. Khaled Elleithy, and Dr. John James. I also would like to thank my brother Mohannad Abuzneid for his support.

I also would like to thank the University of Bridgeport for all it has provided to me. Special gratitude to President Neil Salonen for his support and encouragement. Many thanks to my colleagues, faculty and staff of the school of engineering.

TABLE OF CONTENTS

ABSTRACT iv

ACKNOWLEDGEMENTS v

TABLE OF CONTENTS vi

LIST OF TABLES ix

LIST OF FIGURES x

CHAPTER 1: INTRODUCTION 1

1.1. Research Problem 3

1.2. Research scope 4

1.3. Motivation behind the research 8

1.4. Potential Contributions of the proposed research 8

CHAPTER 2: BACKGROUND AND LITERATURE SURVEY 10

2.1. Anonymity literature survey: 13

2.2. Providing temporal privacy through fake sources 19

2.3. Providing temporal privacy through delay 20

CHAPTER 3: PROBLEM STATEMENT AND RESEARCH PLAN 22

3.1. System Model and Goals 22

3.2. Network Model 26

3.3. Attack / Threat model 30

3.3.1. Passive attacks 30

3.3.2. Active attacks 31

3.3.3. View of the adversary 32

3.4. Traffic model 32

3.5. Implementation and test plan 33
CHAPTER 4: ANONYMITY 34
4.1. Pre-Deployment phase 34
4.2. Setup phase 34
4.2.1. Creating pseudonyms 36
4.2.2. Deleting security information 38
4.3. Communication phase 38
4.3.1. Transmission as a sensor 39
4.3.2. Transmission as a forwarder 40
4.3.3. Acknowledgement 41
4.3.4. Transmission as a broadcaster 46
4.3.5. Limited Broadcast Messages 47
4.3.6. Fake Broadcast Message 48
4.4. SN removal 49
4.5. SN Addition 50
CHAPTER 5: DATA AUTHENTICATION AND INTEGRITY 51
CHAPTER 6: TEMPORAL AND RATE PRIVACY 53
6.1. Simple Global Anti Temporal scheme (SGAT) 57
6.1.1. Security analysis 61
6.1.2. Delivery time 61
6.1.3. Energy cost 62
6.2. Energy Controlled Anti Temporal scheme (ECAT) 63
6.2.1. Changing ω_i from fixed to variable 63
6.2.2. Reducing fake messages and delay for real messages 66
6.2.3. Energy conservation 68

6.2.4. Handling rate attack 70
CHAPTER 7: ANONYMITY AND SECURITY ANALYSIS 73
7.1. Security against passive attacks 73
7.2. Security against active attacks 75
7.2.1. Soft-active attacks 75
7.2.2. Hard-active attacks 77
7.3. Sink security 78
7.4. Link anonymity 80
7.5. Timing privacy 80
7.6. Routing Privacy 81
7.7. Data Privacy 82
7.8. SLP and BLP 82
7.9. Entropy 82
CHAPTER 8: PERFORMANCE EVALUATION 85
8.1. Delay: 85
8.2. Energy cost 88
8.3. Transmission Rate Privacy 91
8.4. Storage Evaluation 95
8.5. Processing and Computational Evaluation 97
CHAPTER 9: CONCLUSIONS AND FUTURE WORK 99
REFERENCES 101

LIST OF TABLES

TABLE 2.1: SOLUTIONS FOR SLP USING ANONYMITY..... 13

TABLE 4.1: REFERENCE OF IMPORTANT PARAMETERS AND TERMS USED BY FAC..... 35

TABLE 4.2: SHARED VALUES AMONG SENSOR NEIGHBORS. 38

TABLE 8.1: PERFORMANCE COMPARISON..... 96

LIST OF FIGURES

FIGURE 1.1: THE NEED TO PROVIDE SLP AND BLP IN WSN. 2

FIGURE 1.2: PRIVACY CATEGORIES IN WSN. 3

FIGURE 1.3: BEHAVIOR OF THE ADVERSARY IN WSN. 7

FIGURE 1.4: ADVERSARY VIEW IN WSN. 7

FIGURE 3.1: CLOSE-LOOP NETWORK IN WSN. 26

FIGURE 3.2: MODULES USED IN WSN 29

FIGURE 4.1: THE SEQUENCE OF A MESSAGE TRANSMISSION FROM SN_l TO THE BS. 42

FIGURE 4.2: USING $APID_l$ FOR ACKNOWLEDGEMENT WITH NO ERRORS. 43

FIGURE 4.3: ACKNOWLEDGEMENT FOR A MESSAGE WITH ERRORS. 45

FIGURE 4.4: HANDLING LOST ACKNOWLEDGEMENT. 45

FIGURE 4.5: SLIDING WINDOW FOR RECEIVED PIDS [7]. 47

FIGURE 6.1: A PROBABILISTIC DISTRIBUTION FOR FAKE MESSAGES [41, 70]. 54

FIGURE 6.2: HAVING MULTIPLE COLLUDING NODES WILL REDUCE SYSTEM ANONYMITY EXPONENTIALLY [6, 41, 70]. 56

FIGURE 6.3: DELAYS INCURRED AT EACH SN [6]. 59

FIGURE 6.4: TOTAL DELAY REQUIRED TO SEND A MESSAGE FROM A SOURCE TO THE BS THROUGH (U) HOPES [6] 62

FIGURE 6.5: ANONYMITY WITH FAKE MESSAGES. 65

FIGURE 6.6: THE GENERATION OF THE SEQUENCE OF INTERVALS ASSIGNED FOR EACH SENSOR. 66

FIGURE 6.7: SN IS ASSIGNED A SEQUENCE OF INTERVALS WHICH REPEAT UNTIL SENSOR LIFETIME ENDS. 67

FIGURE 6.8: ANY NODE ASSIGNED SUBINTERVAL $\omega i = 2$ WILL SEND A FAKE MESSAGE IF IT DOES NOT HAVE A REAL MESSAGE TO REPORT. 69

FIGURE 6.9: HOW TO CHOOSE THE FORWARDING NODE ACCORDING TO THE ENERGY LEVELS OF THE NEIGHBORS. 69

FIGURE 6.10: HIGHER TRANSMISSION RATE NEXT TO THE BS [6]. 72

FIGURE 6.11: TO BALANCE THE HIGHER DATA RATE NEARBY THE SINK, WE ACQUIRE A HIGHER DENSITY SENSOR DISTRIBUTION. 72

FIGURE 7.1: HARD-ACTIVE ATTACK TRIES TO GET THE BS BY INSERTING A SIGNATURE IN THE TRANSMITTED MESSAGE. 79

FIGURE 7.2: PROTECTING THE BS BY HAVING ONION ENCRYPTION. 81

FIGURE 7.3: DEGREE OF ANONYMITY (A). 84

FIGURE 8.1 : TOTAL ACCUMULATED DELAY PER ONE NODE. 86

FIGURE 8.2: FIGURE 22. AVERAGE ACCUMULATED DELAY PER ONE NODE. 87

FIGURE 8.3: THE ACCUMULATED DELAY IS A FUNCTION OF THE HOP COUNT (HC) AND THE SIZE OF ω. 87

FIGURE 8.4: THE ENERGY DISSIPATION INCREASES AS THE NUMBER OF NEIGHBORS AND THE SENSOR TRANSMISSION RANGE INCREASES. 90

FIGURE 8.5: A SIMULATION FOR THE MAXIMUM POSSIBLE FAKE MESSAGES PER SUBINTERVAL USING ECAT. 90

FIGURE 8.6: THE AVERAGE FAKE MESSAGES IN A BUSY NETWORK WITH 70% OF THE SLOTS OCCUPIED BY REAL MESSAGES. 92

FIGURE 8.7: THE SIMULATION SHOWS THE TOTAL REAL MESSAGES, FAKE MESSAGES AND IDLE SLOTS. 93

FIGURE 8.8: SIMULATION FOR BUSY NETWORK WITH DIFFERENT RMIN THRESHOLD VALUES TH=10, 11 AND 12. 95

FIGURE 8.9: SIZE OF STORAGE MEMORY USING DIFFERENT PRIVACY SCHEMES. 98

CHAPTER 1: INTRODUCTION

Wireless sensor networks (WSN) consist of many nodes or hosts called sensors. A wireless sensor device is a simple autonomous host device. It can sense a phenomenon, convert the sensed information into a form of data, process the data and then transmit the data to an aggregator or a sink for further usage or analysis. The sensor host is very limited in terms of memory, storage space, computing power, communication capabilities, and battery lifetime [1-8]. Wireless sensor network (WSN) consists of many sensor nodes, which are distributed in a certain application area to inspect some phenomenon. Sensor nodes will be left unattended in most of the applications until energy of the batteries are completely depleted [9]. Quyan et al. [10] suggest that there are three research areas which require attention in the area of WSN: *(i.)* sensor sophistication and accuracy to fit different applications. *(ii.)* Performance, routing and data delivery, and *(iii.)* Data privacy, security and integrity.

There are many different applications for sensor nodes. However, we are interested in *monitoring and tracking* applications in this work. The WSN consists of many sensor nodes that monitor a certain area and track the presence of certain objects of interest such as an animal in the wildlife, a patient or a doctor in a hospital, or a fellow soldier or a vehicle in the battlefield. When the sensor node senses the object, it will report the data to the sink (or to multiple sinks) either directly or through other neighboring sensors. A sink is a wireless base station (BS) that is much more powerful than a sensor node in terms of

storage capacity, power supply, communication capabilities and computing power. The sink usually has two responsibilities, to control and manage the WSN and to collect (aggregate) the data from the sensor nodes. One of the most common applications discussed in WSN privacy is the *Panda* monitoring game [4, 11]. The system needs to provide source location privacy (SLP) for the sensors. When a sensor node detects a Panda in an area, it should report via a message, which is transmitted through intermediate nodes to the sink. In order to protect the Panda from hunters or adversaries (ADVs), we need to implement in place an efficient source location privacy protocol. In such a scenario, location privacy is much more important than confidentiality of sensed data itself. Source location privacy is even more important in military, homeland security, and law enforcement in addition to many civilian applications [12]. In addition, the WSN needs to provide sink location privacy (BLP) to protect the sink.

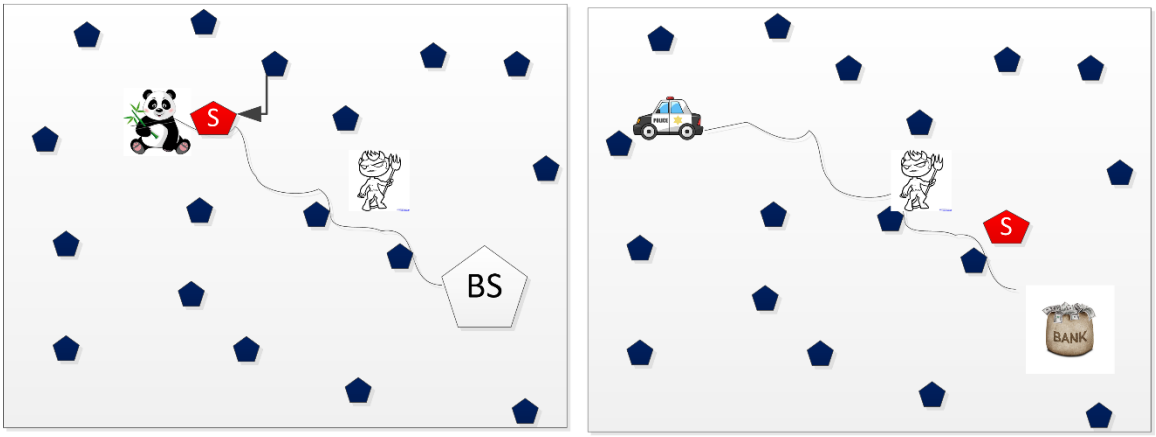


Figure 1.1: The need to provide SLP and BLP in WSN.

Figure 1.1 demonstrates the importance of source and sink location privacy in WSN. In the first application on the left, the hunter is tracking the overheard signals to capture location of the source where he will be able to capture the Panda. In the second

application on the right, the criminal is trying either to get to the asset (source of money) or to capture location of the police patrol / officer either to avoid him or to attack the officer.

1.1. Research Problem

Privacy in WSN is categorized into two categories [13, 14] as presented in Figure 1.2. The first category is *content or data privacy*, which usually focuses on *data aggregation* (sent from sources to the sink) and *data query* (sent from the sink to the sources). It provides for data confidentiality, integrity, freshness, non-repudiation and other privacy issues [4, 14]. The second category is *context or contextual privacy*. It involves location privacy, identity privacy, routing privacy, temporal/timing privacy and transmission-rate privacy. Location privacy involves hiding the location of a source node (SN) and the sink of the WSN. Identity privacy ensures that the identity of a SN remains secret. Routing privacy is to hide the traffic flow. Temporal privacy is concerned with hiding the timing relationship between incoming and outgoing traffic [4, 13]. Rate privacy maintains similar transmission frequency by SNs.

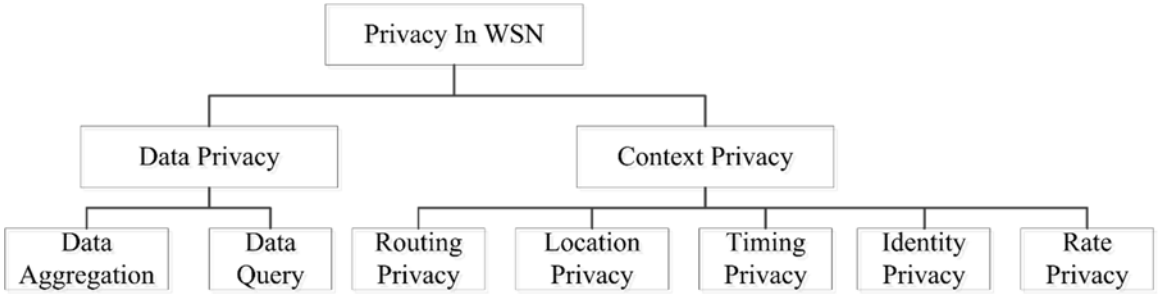


Figure 1.2: Privacy Categories in WSN.

1.2. Research scope

In this work, we shall focus on location privacy, which includes two categories, source location privacy and base-station location privacy. Achieving both source and base-station location privacy is crucial for many applications in WSN. BLP is very important because the sink acts as a gateway to the backbone network, thus it might be a single point of failure, especially if the network has only one sink. In addition, the sink is the controller (or the brain) of the WSN, so if an adversary compromises the sink, it can run many Denial of Service (DoS) attacks or other similar active attacks which could cause the whole network to collapse [13]. The sink also could have very sensitive data that no one can afford compromising, especially if it is a single data aggregator from all the sensors in the network. Compromising the sink could enable the adversary to track the transmission back to the source location. Consequently, BLP is important for the privacy of the SLP itself. SLP is much more crucial in certain applications compared to BLP. It is obvious that the life of the Panda in the habitat or the soldier in the battlefield depends very much on SLP. The existence of *deterministic* or highly *probabilistic* SLP is a core requirement in many applications. We have seen many applications refrain from using WSN due to the weak SLP. Therefore, providing concrete solutions for SLP would lead to using WSN in many future applications. Some of the literature discuss anonymity, untraceability and unlinkability. Pfitzmann et al. [15] indicates that *anonymity* is the state of being unidentifiable within a set of objects. *Untraceability* is the inability of an adversary to trace individual data flows back to the sink or the source. *Ulinkability* is to prevent any adversary from learning the identities of the sources or the sink at the same time.

Providing concrete end-to-end location privacy is a very complicated problem because of the open nature of the transmission media in WSN. The effectiveness of a solution will depend on whether the sensor node is mobile or static. Another important factor is the nature of the adversary. The adversary could be local, multi-local or global. It could be invasive or non-invasive. The variety of complex scenarios and applications is the reason for exploring many solutions for SLP and BLP. Unfortunately, in some solutions, one cannot provide privacy for both SLP and BLP concurrently, which requires a compromise in order to achieve better results for one over the other. We found from the literature that most of the proposed schemes usually focus on either SLP or BLP with significant interest on the SLP problem.

The first work to classify context privacy was done by Kamat et al. [16, 17], where they addressed the Panda hunter game. They claim that the routing scheme is responsible to hide source location of a subject. They have used two metrics to measure SLP: *safety period* and *capture likelihood*. Safety period is the number of messages that a source sends before it is captured. The capture likelihood is the probability that an adversary can capture the source within a certain period.

There are generally, two ways to locate a source using passive attacks: *traffic analysis* [13, 18] and *packet tracing* [13, 19, 20]. The traffic analysis can determine the location of sources or sink by analyzing the traffic. As an example, the volume of packets near the sink will be higher compared to the volume of packets near normal SNs. Packet tracing can be used to find the source location because adversaries may use radio-frequency localization techniques to perform a hob-by-hop trace. The adversary can move quickly

during packet trace. In addition, it works very well to trace mobile nodes due to its fast response compared to traffic analysis [13, 19]. One of the most important factors for designing a scheme or a protocol for SLP and BLP is the nature of threat / attack model and the adversary capabilities. Conti et al. [4] presented a classification for adversarial capabilities. They classified the adversary based on four properties.

1. The behavior of the adversarial node, as presented in Figure 1.3
    a. Access level: Internal or external,
    b. Interference: active or passive,
    c. Compliance: semi-honest or dishonest.
2. View of the network, as presented in Figure 1.4:
    a. Global adversary or laptop class adversary [10],
    b. Local adversary or mote class adversary [10],
    c. Multi-local adversary or semi-global adversary [21].
3. Resources:
    a. Memory,
    b. Data storage,
    c. Computation and processing power,
    d. Device rich: antennas, spectrum analyzers, GPS [22].

4. What the adversary knows about the WSN:

    a. The topology of WSN,

    b. Location of a SN or a sink,

    c. Identity of some SNs or a range of identities,

    d. Routing schemes/algorithms/protocols,

    e. Keys or other encryption functions and information,

    f. Events distribution /schedule,

    g. Other related information to the WSN.

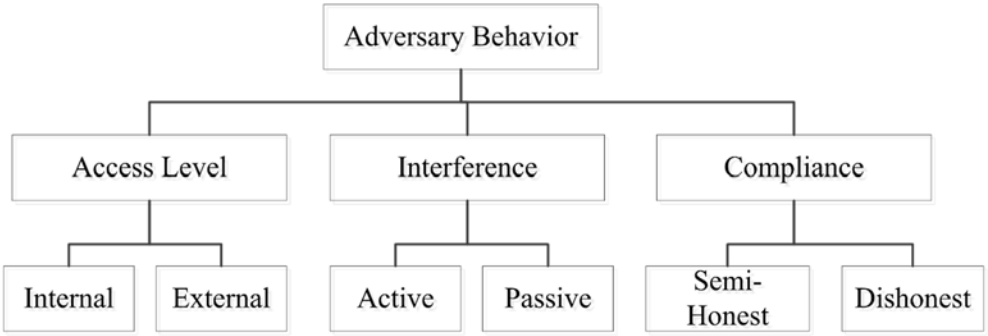


Figure 1.3: Behavior of the adversary in WSN.

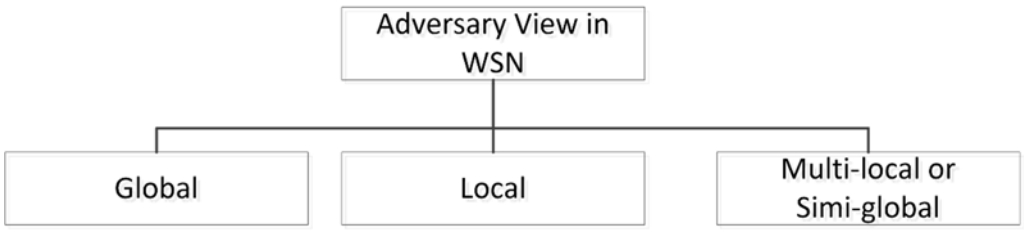


Figure 1.4: Adversary view in WSN.

1.3. Motivation behind the research

Most of the previous research focus on providing either SLP or BLP. Very limited approaches in the existing literature addressed end-to-end privacy, which is a core requirement for many applications. As will be explained in following section, we propose a fortified framework that provides privacy under a strong threat model.

1.4. Potential Contributions of the proposed research

We provide a framework that will be tested against other solutions using the following metrics:

Security and entropy: The probability that the adversary successfully identifies the source, the intermediary SNs or the sink

Energy cost: What would be the cost for an event message to be sent from a source to the sink. This will include both the computational and communicational cost.

Storage and memory cost: What size of memory is needed, considering the fact that the SNs need to store some information about the network, topology and the neighborhood.

Delivery time: What would be the time (or delay) to send an event message from the source to the destination?

Safety period: How long it takes the adversary to capture the first sensor node in the network.

Our proposed framework provides a modular system that is configured for a variety of network models and threat models. The framework will provide anonymity, authentication, temporal privacy and transmission rate privacy.

CHAPTER 2: BACKGROUND AND LITERATURE SURVEY

There are many solutions, which have been presented to solve the problems of SLP and BLP. Li et al. [14] discussed some of the solutions for SLP but they did not aim to create a survey. The comprehensive survey for SLP was presented in the work by Conti [4], where they categorized the solutions into eleven groups. They have discussed many solutions and compared them in terms of power consumption, the attack / threat model, view of the network, exposed information, and efficiency in providing location privacy. They also discussed some issues that each solution exhibits. They have listed the following categories for solutions:

Random walk: trying to counter traffic analysis and hop-by-hop packet tracing by making the path of the transmitted packets completely random. The SN randomly forwards the packet to one of the neighbors based on a certain probability. To enhance SLP, random walk could incorporate other techniques such as dummy messages to improve the safety period [4, 16, 23-32].

Geographic routing: routing the packets to destination according to the geographic information available in the WSN [4, 22, 33].

Delay: exhibiting timing / temporal privacy, which provides location privacy by making timing analysis attack or a hop-by-hop trace attack unfeasible. The adversary cannot correlate the incoming traffic to the outgoing traffic of a SN [4, 34, 35].

Dummy data sources: obfuscate the real data traffic to divert the adversary; some nodes send dummy packets for every real packet sent out. Dummy packets do not contain any useful data. The adversary cannot decrypt either the fake or the real data so it will not be able to distinguish which one is the real packet. Such schemes could work very well to protect against global adversaries [4, 10, 11, 16, 36-51].

Cyclic Entrapment: Many nodes act as fake data sources, which form a loop inside the WSN. Fake traffic loops in a cyclic route with no decided start or end for the loop. This will confuse any local adversary within these loops and makes it almost impossible to trace back hop-by-hop to the source [4, 52, 53].

Anonymization: Having the ID's of SNs anonymous makes communication untraceable and in some case unobservable. The anonymization could be achieved by either aggregation or pseudonyms [12, 24, 28, 54-58].

Cross-layer routing: This scheme will use sub layers below the network layer for communication; such as using control messages from the medium access control (MAC) layer. Such techniques are very useful also to enhance energy consumption. Unfortunately, they are good against local adversaries only presuming the adversary does not have the ability to inspect lower level messages such as beacon frames [4, 59].

Separate path routing: Having a separate route for packets will complicate hop-by-hop tracing because the packets of one SN do not follow the same route. The adversary cannot hear all the packets due to its limited hearing range or due to being far from the source. The adversary needs to listen to multiple packets before it can locate the source while it is moving hop-by-hop towards the source [4, 60].

Network coding: SN not only originates or receives a packet, but it is allowed to do some computations on the packet before sending it out to the next hop. The computation could imply encoding/re-encoding. The input and output packets are unlinkable which makes it harder for the adversary to do a traffic analysis attack [4, 61, 62].

Limited node detectability: These schemes work on the physical layer of the WSN to hide the location of the SN or the presence of transmission. One way is to silence the SN so it is much harder for the adversary to locate it. Another way is to lower the radio transmission power so that the local adversary could not hear the transmission [4, 63-67].

There are many other solutions, which could entail multiple schemes to achieve SLP or BLP. Each solution usually presumes a certain network model, threat model, and routing model. One cannot find a standard network model, routing model or threat model. Most of the solutions presume static SNs and very few tolerate mobility. In addition, the topology is not standardized so some work assume tree topologies and some others assume clustered topologies. This diversity in the pre-setup of the network makes it hard to compare the outcomes of the provided solutions for location privacy and anonymity.

2.1. Anonymity literature survey:

Anonymity is an old issue that was discussed for mobile networks, Ad Hoc networks and, Internets. Recently, it has become a concern for WSNs. We have identified most of the literature discussing solutions for anonymity in WSN. We have included them chronologically in Table 2.1.

Table 2.1: Solutions for SLP using anonymity.

No	Scheme	View of the adversary	Anonymity technique	Passive attacks	Active attacks
1	SAS & CAS [54]	Global	Pseudonyms	Eavesdropping, SN compromise, limited traffic analysis	-
2	HIR & RHIR [55]	Global	Pseudonyms	Eavesdropping, SN compromise,	-
3	APR [56]	Local	Pseudonyms	Eavesdropping, hops-tracing	SN compromise
4	DCARPS & Global DCARPS [12]	Global	Pseudonyms	Eavesdropping, hops-tracing	-
5	ACS [28]	Local	Pseudonyms	Rate monitoring, time correlation, identity analysis, hops-trace	-
6	MQA [57]	Global	Aggregation	Eavesdropping, hops-tracing	Packet injection
7	PhID [4, 58]	Local	Pseudonyms	Eavesdropping, traffic analysis	-
8	EAC [68]	Global	Pseudonyms	Eavesdropping, traffic analysis	DoS, SN compromise, Traffic injection

Misra et al. [54, 69] introduced two solutions for anonymity using pseudonyms in order to counter the traffic analysis of a global adversary in WSN. The first scheme is called

the simple anonymity scheme (SAS) [4], where SNs communicate using pseudonyms. A large pseudonym space is created and shared by all the SNs. Every SN will get a subspace before deployment. The sink knows which subspace belongs to which SN. The WSN consists of clusters where each cluster has a cluster head and SNs.

A SN stores two ranges of pseudonyms for each neighbor, one range for uplink (towards the sink) and the other range for downlink. The SN uses *index i* within its table to know which range is used for that specific neighbor. The pseudonym is mapped to the shared key used for encryption and decryption of the messages. When a SN wants to send a message to a neighbor within the cluster, it chooses one of the outgoing pseudonyms concatenated with *index i* as the destination, which is also concatenated with the message encrypted using the shared key mapped to the used pseudonym in the table.

SAS uses high memory to store pseudonyms so another scheme was introduced to reduce the amount of storage used at the expense of a higher computation for pseudonyms. The scheme is called cryptographic anonymity scheme (CAS) [69]. In CAS, nodes use a keyed hashed function to generate the necessary pseudonyms. To generate a pseudonym for a neighbor, SN uses the shared key with a seed known by both the SN and the neighbor to generate the necessary pseudonyms.

Ouyang et al. [55] introduced two solutions for SLP. The first solution is hashing-based ID randomization (HIR) which is a solution for CAS in case a SN is compromised. The SN communicates with a neighbor using the hash value (HV) of the neighbor. A SN creates HV for each neighbor. It uses a keyed hash function. It also creates a HV to

communicate with the sink, based on the SN's ID and a pairwise shared key (shared with the sink). The HV will be renewed by key-hashing the current HV with the shared key.

The second solution from Ouyang et al. [55] is called reverse hashing based ID randomization (RHIR). SN's generates all the hash values, HV's, and store them as a chain. SN's use the HVs in reverse order and use the HV's to communicate in the same way as in HIR. When a HV is used, it will be dropped out. HIR and RHIR provide SLP in a dense network.

Sheu et al. [56] introduced Anonymous Path Routing (APR) which works against a global adversary. It hides the identities of the nodes and changes the encryption of each message at every hop. It uses three procedures: anonymous one-hop communication, anonymous multi-hop path routing and anonymous data forwarding.

Nezhada et al. [12] introduced destination controlled anonymous routing protocol for sensor networks (DCARPS). It works to handle a local adversary under dense network only. It consists of five phases. In the first phase, every SN gets a unique ID, a random *nounce* DL shared with the sink, and a secret shared key *k,* which is shared between the SN and the sink. In the second phase, the sink learns the topology of the network. In the third phase, this sink calculates a tree that covers the network, with the sink as the root of the tree. The sink assigns labels for uplink communications to each of the SN's. Each SN gets one outgoing and one incoming label. If a SN needs to send a packet towards the sink, it shall use its outgoing label embedded within the packet. The neighbor uses the label if it matches its incoming label to accept the message and then transmit it again towards the root (the sink) using its outgoing label. In the fourth phase, the sink fixes the routes. The

sink sends out a cryptographic onion message, which consists of multiple encrypted layers. Whenever the SN overhears a broadcast onion over the network, it checks whether the onion contains the SN's DL as a destination. If that is the case then it accepts the onion message and then uses the new DL as the label and rebroadcast the message again. The onion message is kept forwarded until every SN has its label. In the fifth phase, the sensing and reporting shall start. The SN encrypts the data by the shared key with the sink and appends its own outgoing label to the packet. On the way to the sink, Each SN receives the packet and re-encrypts the data using the key it shares with the sink, and adds its own outgoing label and transmits the packet onward. This process is repeated until the packet arrives at the sink. The protocol defines how the sink can send command messages using downlink labels. Each SN uses a random back-off period to prevent collection before it tries to transmit the packet.

Probabilistic DCARPS is similar to regular DCARPS, however, it has two versions. The SNs may have multiple outgoing labels and multiple paths towards the sink where each path has its own *PathID*. One version allows every intermediary SN to select a random path with a *PathID* to forward the packet to the sink, while in the second version only the source selects a certain *PathID*.

Luo et al. [28] introduced a solution against a local adversary, called anonymous communication scheme (ACS). Each SN shares an individual key with each of its neighbors. The SN and its neighbor use a hash function with the shared key and the real ID's of both to calculate the outgoing and incoming hidden identities. The hidden identities are updated periodically using the shared key. In addition, each SN stores a table with

multiple columns. The first column in the table contains the identity of each individual neighbor. The second column contains the sequence of the hashing, which indicates how many times the hash function is applied to the identity. The third column contains the current outgoing hidden identity of that neighbor. The fourth column contains the current hidden incoming identity of that neighbor. The SN can send a message to its neighbor after it encrypts the data with the shared key and addresses the packet to the incoming hidden identity of the neighbor.

Di Pietro et al. [57] introduced max query aggregating (MQA). This method does not use significant computational overhead because it is based on data aggregation. It presumes that the network needs to report the maximum value. This also assumes the network is built as a hierarchical tree where the sink is the root of the tree. Every SN has a unique symmetric key shared with the sink. The SN senses data, which is an integer of *i* bits. The WSN goes through *i* rounds to gather the sensed value from all the SNs.

Park et al. [58] introduced a solution called Phantom ID (PhID [4]), which is less computationally intensive compared to ACS. Each node has a real ID abbreviated as *SID* and a phantom ID abbreviated as *PID*. SNs share a pair-wise key with the neighbors. When a sensor is deployed, it uses two pre-distributed parameters, *q* and *p*, to calculate its *PID* using: $PID = q^{SID}$ mod p. Every SN shares its PID with its neighbors encrypted with the pairwise shared keys. Every SN stores a table, which contains the following about the neighbors: the neighbor's PID, the neighbors SID, the direction; uplink or downlink. The SN exchanges its table with its neighbors. All the communications will use PID with a hidden vector, *HV*. The *HV* consists of the SID of the sender XOR-ed with the shared key

of sender and receiver, and XOR-ed with a random number, which both sender and receiver can generate. HV is updated regularly. When SN sends a message, it combines its PID, the HV, a timestamp, the destination's ID, the encrypted data and a modified SMAC code in a message and sends it towards the sink. The sink uses PID, HV, and the adapted SMAC to check the authenticity and integrity of the message. SNs regularly update their PID and HV to further improve privacy.

An important solution against a global adversary introduced by Chen et al. [68] called efficient anonymous communication (ECA) provides send, link and sink anonymity. The solution consists of three phases. The first phase is the network pre-deployment, in which each SN gets preloaded with parameters α and β, two hash functions $H1$ and $H2$, and a unique ID. The second phase is the network initialization. Each SN gets to know its location and the distance to the sink by metric of the number of hops. Every SN creates a global anonymous identity (AI) based on the hash function H1, the parameter α, and the node's unique ID. Then the node creates an anonymous broadcast identity *BAI* base on the hash function *H1*, the parameter β, and the node's ID. Then, it calculates an anonymous one-hop identity (*OHAI)* and an anonymous acknowledgement identity (*AAI)*. At the third phase, SNs start sensing and reporting data. A source SN chooses one of the neighbors and sends the data with the shared *OHAI* address of that neighbor. The data will be encrypted with the shared key between the source and the neighbor. The data will also be encrypted with the key shared between the source and the sink. The sink will recognize the source by its *AI* identity included in the message. *AAI* is used for acknowledgement. *OHAI* will be updated after each successful transmission. *BAI* is used for anonymous broadcasting.

2.2. Providing temporal privacy through fake sources

We know that anonymity is not enough to achieve a fortified end-to-end privacy. There are some solutions based on fake data sources where SNs send out fake packets to other nodes within the network. Some literature call them dummy packets. A fake packet does not contain any real information about any real events but it helps to obfuscate the real traffic and to divert the adversary by mimicking the presence of a fake source. The literature shows reasonable solutions using the fake source. Some of them are designed to handle local adversary and some of them are suitable for global adversary. Some of the literature presume a certain routing scheme, topology, or network and threat models. Some of the literature used in this work are explained hereafter.

Alomair et al. [70] presented two algorithms: A-real and A-fake. A-real is used when there are real events in the WSN. In this case, SNs embed real traffic within fake traffic, based on statistical tests. A-fake is used when the SNs do not detect any real events. In this case, the SNs presume the presence of fake subjects by presuming fake events, which they are then embedded within the dummy traffic. The adversary will see no difference between A-fake and A-real.

Ouyang et al. [10] introduced three different solutions to handle the global adversary problem. The first solution is the globally optimal algorithm (GOA). Each SN has a pseudo random number generator that defines the interval time. If the SN has data, it will send it at the end of the time; otherwise, it will send a fake message. SNs share the interval times for all the nodes in the WSN so it can calculate the shortest path to the sink.

The SN needs to know the complete topology of the network, which is very memory consuming.

The second solution by Ouyang et al. [10], is the heuristic greedy algorithm (HGA) where SNs follow the same procedure as in GOA except that the SN does not know the complete topology, but it only has the information of the location and the seeds of its neighbors. The source SN can calculate the best node among the neighboring SNs to forward the packet to it.

The third solution by Ouyang et al. [10] is the probabilistic algorithm (PBA) where nodes still follow the procedure of HGA, except that they don't send fake messages at the end of every interval. It uses probability *p* to decide whether to send a fake message or not. The value of *p* will reduce the communication overhead at the expense of SLP.

2.3. Providing temporal privacy through delay

We can enhance SLP and BLP by having temporal privacy, which creates hop-by-hop trace attack or timing analysis attack. It works very well for the monitoring of moving objects [4]. There is some literature that addressed this using issuing packet delay techniques.

Hong et al. [34] introduced probabilistic reshaping (PRESH) to counter the adversary that uses timing analysis techniques. A SN shares a key with its neighbors for encryption. Packets are padded to have the same size. The identity of the source SN is included in the header without encryption while the identity of the receiver is hidden. When a source SN_i has a packet to send, it sends the packet directly to an intermediary SN_j where

it delays the packet before it forwards it again to the next hop. The delay is random and follows an exponential distribution, which can be set based on the protocol parameterμ. Received packets will be queued and sent in arbitrary order due to the random delay. The adversary cannot tell after the queued packet was sent, the source of the packet because the network will have many packets transmitted during the delay time. The setup of parameter μ is used to adjust between timing privacy and deliver latency. If a node has only one packet in its buffer while there are no other transmissions in the neighborhood, then transmitting it will give the adversary 50% chance to guess if the packet was sent by a source or intermediary node. Hong et al. [34] introduced and upgraded PRESH extended probabilistic reshaping (exPRESH) to counter such a scenario. The SN will delay the packet in its buffer again up to *D* time. Thus, the original exponential delay should be introduced in PRESH and the delay introduced by exPRESH.

Kamat et al.[35] introduced rate controlled adaptive delaying (RCAD). The payload is encrypted; however, the header is a plaintext. The header includes the ID of the source SN and the ID of the last intermediary SN as well. Every SN should delay its packets for a random amount of time after which the packet is sent to the next hop. The delay will vary in every node according to a statistical distribution. The intermediary SN fills up its queue and delay the transmission for variable delay periods. The nodes also need to count for the fact that the buffer could be filled, which could cause data loss.

CHAPTER 3: PROBLEM STATEMENT AND RESEARCH PLAN

3.1. System Model and Goals

In this work, we will provide a framework for End-To-End location privacy and anonymity. In any WSN, many SNs are distributed in the application area. The SNs sense some condition and then report the data back hop-by-hop to the sink, which we shall call it a base station (BS), hereafter, in this work. In this work, we will design a framework that provides the following security elements:

1. Sender anonymity,
2. Receiver anonymity,
3. Link anonymity,
4. Source location privacy (SLP),
5. Base-station/sink location privacy (BLP),
6. Timing privacy,
7. Rate privacy,
8. Route privacy,

9. Data privacy,

10. Safety period and,

11. Energy preservation.

We take that the sorted priorities to achieve are as listed above. Thus, sender anonymity has a higher priority than receiver anonymity, which in turn has a higher priority than SLP and then BLP and, so on. There are many design constraints, one of which is the limitation in all the resources of the SN. This should include the battery lifetime, CPU capabilities, memory space, and storage capacity. We will consider all of the above limitations and we take into consideration the need to be very conservative and efficient. Anonymity in WSN aims to prevent any adversary from knowing the identities of the sender or the receiver prior to, during, and after the communication has taken place. This should include the source SN where it senses the phenomenon and sends data out, and the sink, which receives the data from the source SNs. The sink checks data integrity and correctness, and forwards it to the core network for further processing. Security of the sink is crucial for the security of the whole WSN. It works as the network controller and has other critical roles in the network. Security of the source SN in some applications is also more important than the sink's, since any disclosure of its identity or location could be devastating, especially if that were to endanger the life of a soldier, a patient or an animal (the Panda game). Node anonymity is a complex situation that involves the *sender* anonymity, the *receiver* anonymity and the *link* anonymity. Without the triple anonymity, the adversary (or colluding adversaries) could compromise one or more SNs and possibly the base station.

Anonymity is crucial for SLP and for BLP, but it will not be enough, especially with the presence of global or colluding adversaries. The lifetime of a WSN is usually considered the lifetime of the first SN in the network that runs out of battery [12]. Thus, it is very important to have a fair distribution for the SNs.

We have discussed in the introduction many techniques that could be used for privacy. Applying each of them or a bundle of them could help in increasing the security of the network, in particular, location privacy. It will also increase the safety period. However, our major concern is conserving the power consumption of the SNs to increase the overall lifetime of the WSN. Every extra bit of a packet sent will reduce the lifetime of the sensors. Unnecessary processing by the sensor such as encrypting, decrypting, hashing, formatting, error check, acknowledgement, fragmentation etc., will consume more energy. We also know that transmitting data consumes much more than processing the data. We should also plan to distribute the work on all the sensors, especially the intermediary nodes (forwarders), so that we can guarantee the longest possible lifetime and coverage for the spread area of the network (routing). Applying a complex set of routing, key management, encryption, acknowledgement, packet walk, phantom nodes, delay, fake packets, etc., could help in improving the security, for sure, if they are designed to work together seamlessly. Depending on the adversary model and on the security level required, we can adopt a modular framework. As an example, in a relatively safe environment, we could apply light schemes and in a harsh environment, we can apply very sophisticated schemes. A relatively safe environment, here, refers to when the adversary is not very sophisticated and has limited capabilities compared to fully capable adversary in the harsh environment.

If we opt for a better lifetime of the WSN over the safety period, we can use light schemes, and if we need the most to provide security, then we can apply complex and fortified schemes. The decision should be made by the sink, the network brain. However, the SNs can report the condition of the network to the sink. The sink could accordingly apply reasonable configurations to achieve proper privacy and anonymity measures. That is, we adopt the closed loop system.

We would like to divide our framework into multiple modules:

1. Anonymity module,
2. Data authentication and integrity,
3. Temporal and rate privacy module, and
4. Modular network.

Each module can be applied with one or more modules depending on the network requirements. The system should have a bunch of inputs such as the nature of the adversaries in the network, the residual power in the nodes, the desired lifetime or safety period. Figure 3.1 illustrates the closed-loop form for the network.

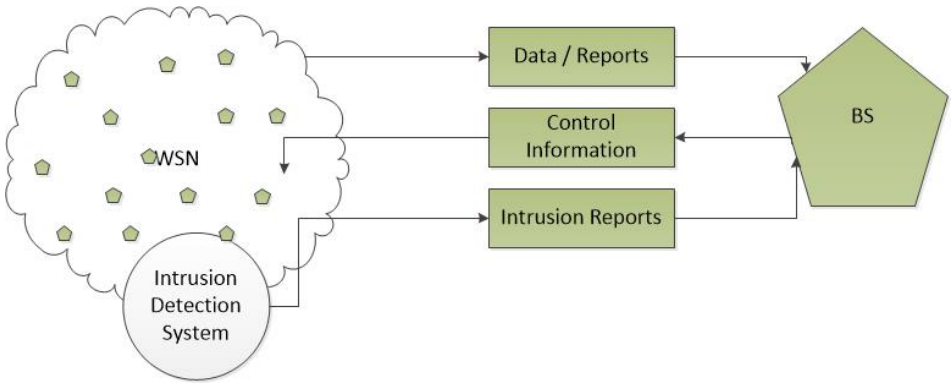


Figure 3.1: Close-loop network in WSN.

3.2. Network Model

The WSN consists of tens to hundreds of small wireless sensors that are randomly distributed in the application area. The sensors are usually similar, low cost, battery driven, limited in memory, exhibit low computation capabilities and *immobile*, and they are not tamper resistant [54]. We assume bi-directional links where two nodes are considered neighbors if and only if they can hear each other [12].The network considers one sink which collects / aggregates the sensed data (stimuli) from all the SNs. The sink works as an interface for WSN to the wired network [54]. We presume that the sink has unlimited power, enough memory, plentiful storage space, and great processing capabilities. Data packets generated by SNs are ultimately destined uplink to the sink and never destined to another SN. However, it could go through a multi-hop-path. Control packets can be sent from the sink, downlink, to the SNs by unicast or broadcast. To enhance BLP, the sink acts like any other SN in the network when it communicates with other SNs to make it indistinguishable. However, it is connected to a wired backbone network, which looks seamless to other SNs and to the adversaries.

Most of the literature show that the operation of WSN network goes through two or more phases. Nezhad et al. [12] provided six phases as an example. However, generally speaking, the WSN runs in three phases: pre-deployment phase, setup phase, and communication phase. During the pre-deployment phase, SNs are preloaded with all the keys, hash functions, IDs and other information needed to setup the network. It also makes sure the batteries are fully charged. Then the setup phase starts right after deployment of the SNs. We presume some safe period where SNs do some calculations and exchange some information required during the communication afterwards. In the communication phase, SNs sense data and send packets hop-by-hop to the sink. We assume that the SNs have the ability to obfuscate the addresses at the MAC level header [54, 71]. All sensors are loosely time synchronized [54].

If we are to seek end-to-end privacy, then the WSN will need a protocol for *network topology discovery* that allows the BS to view the global topology of the network without revealing the sink location [12]. The network protocols should aim to conserve energy to extend the lifetime of the network. An excessive use of security techniques or extra packet transmission would not help the problem of energy conservation. The network routing should achieve correctness, simplicity, robustness, stability, fairness, optimality and efficiency. As general framework for any WSN could include:

1. Pre-deployment module,
2. SNs distribution,
3. Location discovery service,
4. NTDP: Network topology discovery,

5. Link establishment:
    a. Tree,
    b. Clustered network,
6. Key distribution module,
7. Set-up module,
8. Communication module:
    a. Unicast,
    b. Multicast,
    c. Broadcast
    d. Acknowledgement,
    e. Fake versus real messages
9. Data security module,
10. Privacy module for SN's and the sink:
    a. Location Privacy,
    b. Identity privacy,
    c. Routing privacy,
    d. Timing privacy,
11. Routing module,
12. Node revocation and addition,
13. Intrusion detection module,
14. Mobility,
15. Data aggregation,
16. Synchronization:

a. Synchronized network,

b. Time-loose network,

17. Other modules.

These modules and sub-modules interchange and interact depending on the design of the WSN. Some of the modules could be null in certain networks as well. For example, if there are only static SNs, then the mobility module does not exist. See Figure 3.2 for related modules in our proposed framework.

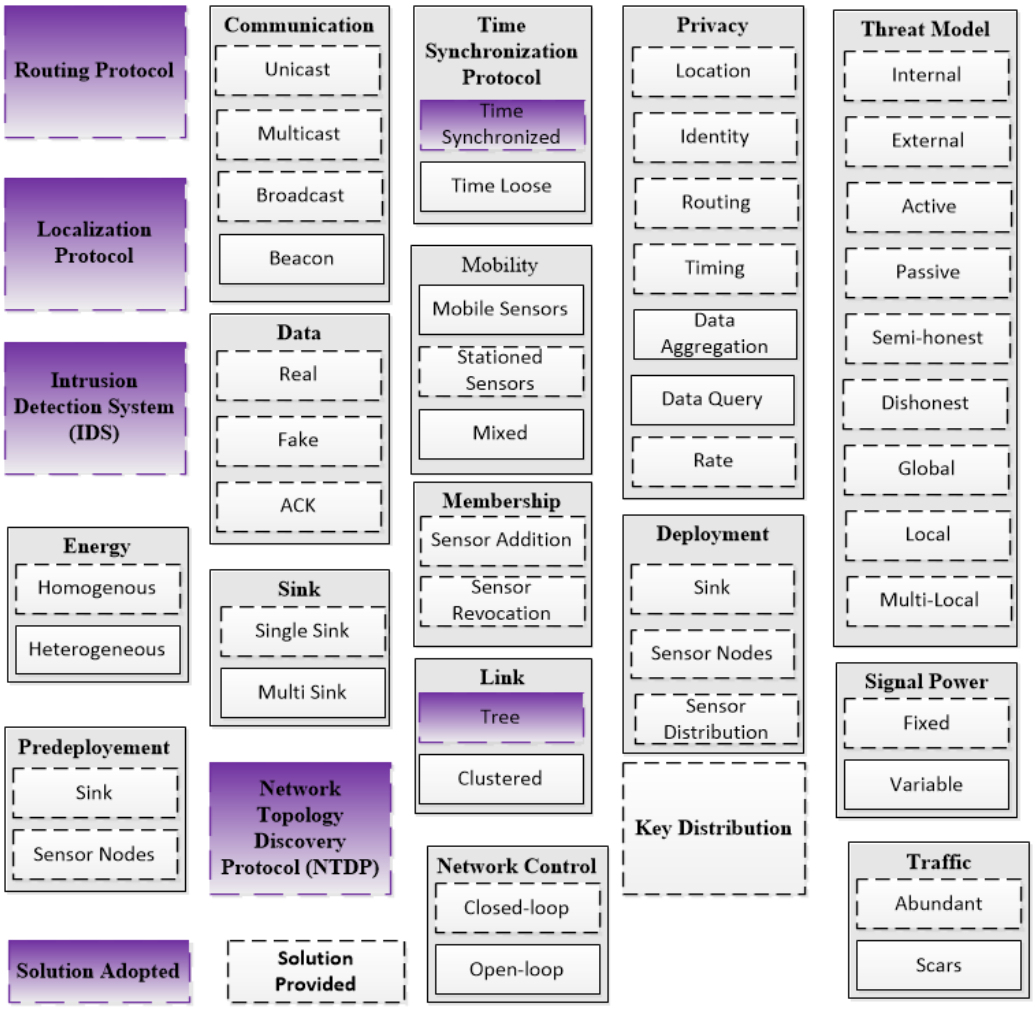


Figure 3.2: Modules used in WSN.

3.3. Attack / Threat model

Any solution for privacy schemes will very much depend on the nature of traffic generated in the WSN and on the sophistication of the adversary trying to attack the network [12]. Our goal is to make it very hard for any adversary to identify privacy information about any packet sent from source to sink. The adversary nodes have very strong capabilities compared to SNs. They are resource-rich, having sufficient energy supply, good computation / processing capabilities, and sufficient storage memory. An adversary could run both passive and active attacks. We presume *Kirchhoff's Principle* [72] for our framework, where the adversary knows everything about the system except the keys and IDs. The threat model will be able to launch both passive and active attacks.

3.3.1. Passive attacks

1. Eavesdropping: The adversary can overhear the messages but cannot decode them.
2. Hop-by-hop trace: The adversary can trace back the packets by overhearing the messages until it gets to the source of the messages.
3. Size correlation: It is a traffic analysis by trying to understand the relation between incoming and outgoing traffic of a SN based on the size of the packets.
4. Time correlation: It is a traffic analysis by using the timing information of related traffic between two SNs and then finding a path from source to BS.
5. Identity analysis: The adversary tries to link two or more overheard IDs to SNs.
6. Rate monitoring: It is traffic analysis where the adversary looks for higher transmission rates as SNs get closer to source or BS.

7. Angle of arrival (AoA): This attack requires special hardware to measure the angle of arrival to find out which direction a transmitting SN lies.
8. RSS: It requires a special hardware for the adversary to measure the signal strength that the adversary receives from the transmitting SN. The signal strength could be used to calculate the distance between the adversary and the SN.
9. Content analysis attack: The length of the control packets and data packets must be a constant and equal on all the links in order to stop content analysis attack [12]. It is even better if there is a random data packet format; that means similar packet length but not repeated format after each hop.

3.3.2. Active attacks

1. Denial of service (DoS): Adversary blocks all further communications in SNs.
2. Replay attack.
3. Forging attack.
4. Active node compromise: The adversary uses the compromised SN to influence the protocol or to detect the presence of other nodes. Adversary also can destroy the SN and remove it from the network.
5. Packet alternation: The adversary can intercept a packet, alter it and send it to the destination.
6. Packet dropping: The adversary can drop packets.
7. Packet injection: The adversary can insert unreal packets into the WSN.

We presume that only few compromised nodes could exist at one time due to implementing a protocol to detect compromised SNs. There are mechanism to remove compromised nodes from WSN [68, 73-76].

3.3.3. View of the adversary

We assume a global adversary, which can monitor the traffic of the entire network and can determine the node responsible for the initial transmission to report the event to the sink. Assuming a global adversary means:

a. Worst-case scenario for area coverage: where colluding sensors can cooperate to cover the whole network area [70].
b. Worst-case scenario for timing: The coverage area of the adversary is not known at any time [70].

We also assume that the adversary is capable of observing SNs transmissions over extended periods. It is not, however, able to break the encryption algorithms used for securing data during transmission.

3.4. Traffic model

Depending on the application of the WSN, the traffic model could vary and thus the solution model needs to be adjusted accordingly. Nezhad et al. [12] discussed this issue and indicated that their proposed DCARPS scheme is designed for busy networks but also provided some solution for light networks. Some networks have *abundant traffic* where sensors detect and transmit many packets such as in environment tracking and monitoring

applications. Such networks can resist global eavesdroppers. Having abundant traffic in the network where many sources send data at one interval of time, makes it harder for the adversary to distinguish a specific data flow among others. The greater the data flow path, the harder it is for the adversary to decipher the identity of the sender. Some applications such as military monitoring, recovery operations and law enforcement provide few data streams during the normal operation. If there is one data flow path in the network, regardless of the routing scheme used, a global adversary can monitor all the packet transmissions and can detect the source as it generates any packet. It is very difficult to protect a *scarce traffic* WSN against a global eavesdropper [12]. In this work, we presume abundant traffic.

3.5. Implementation and test plan

As we have indicated above, the operation of the WSN goes through three phases, which we will discuss in details in the following sections. Our system is designed to achieve the security goals explained earlier. Our goal is to provide a maximum lifetime of the network and improve the safety period of the network as well. Optimally, we provide the six-element security scheme with no security breach for the maximum possible lifetime of the network. The framework is a closed-loop system. Depending on the design used, the lifetime of the batteries in the system will vary. The suggested system model needs to be comparable to the previous security systems in terms of energy consumption but also provide better security, in particular, SN anonymity, SLP and BLP. We have deigned the three modules and we did extensive simulations to test the performance of the framework.

CHAPTER 4: ANONYMITY

The communication process is divided into three phases, namely: Pre-deployment phase, setup phase and communication phase.

4.1. Pre-Deployment phase

Prior to actual distribution of the SNs in the field of application, the SNs need to be tested, fully charged, and preloaded with some parameters. We will use subcase letters ***i*** and ***j*** to describe source and intermediary nodes consecutively. We will use *BS* to describe the sink or the base station. Table 4. summarizes all the parameters and terms used in this work.

4.2. Setup phase

It is typical to presume the WSN is considered secure for some short period after the deployment of sensors and before the steady communication phase. Zhu et al. [77] presented that WSN has a lower bound on the time interval (T_{min}) before the adversary is able to compromise a SN. During this time, the sensors can communicate and exchange all needed information safely. The sink needs to know the location of all the SNs participating in the WSN. Likewise, the SNs need to know their relative locations to the sink and to their neighbors. There are many *localization schemes* which, are proposed in the literature [54, 68, 78, 79]. We presume the network will adopt one of the available efficient localization schemes. Localization allows each SN to know its smallest *hop-count* to the BS $(HC_{i \leftrightarrow bs})$.

Table 4.1: Reference of important parameters and terms used by FAC.

Notation	Definition	Source
ID_i	ID of sensor i	Preloaded
a_i	Random number shared between SN_i & BS	Preloaded
b_i	Random number shared between SN_i & neighbors	Preloaded
c_i	Random number shared between SN_i & neighbors	Preloaded
H1	Hash function to create pseudonyms and the keys	Preloaded
H	Hash function to create data digest	Preloaded
$k_{i\leftrightarrow bs}$	Pair-wise key shared between SN_i & BS	Preloaded
kb_i	Broadcast key for SN_i	Preloaded
fkb_i	Fake broadcast key for SN_i	Preloaded
N	Number of SNs in WSN	Learned
N_i	Number of neighboring for SN_i	Learned
$HC_{i\leftrightarrow bs}$	Hop-count between SN_i & BS	Learned
PID_i	Pseudonym ID shared between SN_i & BS	Calculated
$BPID_i$	Broadcast pseudonym ID	Calculated
$a_{i\leftrightarrow j}$	Random value shared between SN_i & SN_j	Calculated
$k_{i\leftrightarrow j}$	Pair-wise key shared between SN_i & SN_j	Calculated
$OHPID_{i\leftrightarrow j}$	Pseudonym ID shared between SN_i & SN_j	Calculated
$APID_i$	ACK pseudonym ID for SN_i	Calculated
$FBPID_i$	Fake broadcast pseudonym ID	Calculated
T_i	Table in SN_i for shared parameters	Calculated
TIME_STAMP	Time stamp	Calculated
SEQ_NO	Sequence number for a message	Calculated
TTL	Time to live	Calculated
MCG_LGTH	Message size	Calculated
$\boldsymbol{\Delta_{residual}}$	Residual energy	Calculated
$\oplus$	XOR Operation	Operation
$\|\|$	Concatenation operation	Operation

4.2.1. Creating pseudonyms

The key idea is to use pseudonyms instead of using real IDs for the SNs and the BS during communication. Therefore, one disposable pseudonym per one transmission is used. This way, the ADV cannot trace back to the source using multiple messages containing the real ID. There are five kinds of transmissions that could happen in the WSN: (i) Multi- hop transmission between SN_i and BS, (ii) transmission between two sensor neighbors *i* and *j*, (iii) broadcast sent by SN_i or BS, (iv) acknowledgement, and (v) fake broadcast. The process starts by creating a pseudonym ID for each SN_i, we call it for short (PID_i) which is computed using expression below:

$$PID_i = H_1(ID_i \oplus a_i) \qquad (4.1)$$

The SN_i can calculate the broadcast pseudonym ID ($BPID_i$) according to the expression below:

$$BPID_i = H_1(ID_i \oplus b_i) \qquad (4.2)$$

The SN_i can calculate the fake broadcast pseudonym ID ($FBPID_i$) according to the expression below:

$$FBPID_i = H_1(ID_i \oplus c_i) \qquad (4.3)$$

SN_i should, by now, know its entire neighbor set (N_i). SN_i will send a broadcast discovery message ($M_{discovery}$), to exchange parameters with all one-hop neighbors. The format of the message is stated in the expression below:

$$M_{discovery} = K_{dis}(TTL \,||\, ID_i \,||\, k_{i\leftrightarrow bs} \,||\, kb_i \,||\, fkb_i \,||\, a_i \,||\, b_i \,||\, c_i \,||\, \Delta_i \,||\, HC_{i\leftrightarrow bs}) \qquad (4.4)$$

Where *TTL* should be 1 for this transmission. K_{dis} is a shared common encryption key to secure the discovery message. SN_i will receive also a similar broadcast message from SN_j and from all other neighbors. Both SN_i and SN_j will calculate a new random value ($a_{i\leftrightarrow j}$) according to the expressions below:

$$a_{i\leftrightarrow j} = H_1(ID_i \oplus ID_j) \qquad (4.5)$$

Both SN_i and SN_j will calculate also a new pair-wise key $k_{i\leftrightarrow j}$ according to the expression below:

$$k_{i\leftrightarrow j} = H_1(k_{i\leftrightarrow bs} \oplus k_{j\leftrightarrow bs}) \qquad (4.6)$$

SN_i also calculates broadcast pseudonym ID for SN_j ($BPID_j$) according to expression (2) since SN_i has already received the values of ID_j and b_j through $M_{discovery}$. It also calculates the one-hop pseudonym ID ($OHPID_{i\leftrightarrow j}$) shared between SN_i and SN_j as expressed in the expression:

$$OHPID_{i\leftrightarrow j} = H_1(a_i \oplus a_j) \qquad (4.7)$$

Finally, acknowledgement pseudonym ID for SN_i ($APID_i$) will be calculated according to the expression:

$$APID_i = H_1(ID_i) \qquad (4.8)$$

SN_i will create a table (T_i) which contains the shared values with the neighbors as listed in Table 4.2. In conclusion, we have replaced the ID with *quintuple pseudonyms* to reference the SN during the communication.

4.2.2. Deleting security information

After storing all required pseudonyms, parameters and keys in T_i, it would be the time to delete all unnecessary information for the purpose of security [7]. In addition, it will release some memory storage space [68, 80]. Most importantly, SN_i will delete ID_i and $HC_{i\leftrightarrow bs}$, which could be critical information for the adversary. In addition, SN_i shall delete all discovery messages.

Table 4.2: Shared values among sensor neighbors.

Information in T_i per each neighbor	**Tuple for SN_j**
Shared random number	$a_{i\leftrightarrow j}$
Shared broadcast random number	b_j
Shared fake broadcast random number	c_j
Shared broadcast key	$BPID_j$
Shared fake broadcast key	$FPID_j$
Shared one-hop key	$k_{i\leftrightarrow j}$
Current one-hop pseudonym ID	$OHPID_{i\leftrightarrow j}$
Link direction	$link_{i\rightarrow j}$
Residual energy level	Δ_j

4.3. Communication phase

During the communication phase, when sensing and sending data to the BS takes place, there are seven operations that continue until network lifetime ends. These

operations are: (i) Sense and send a message to a neighbor, (ii) forward a message hop-by-hop, (iii) broadcast a real message, (iv) acknowledgement, (v) broadcast a fake message, (vi) SN removal, and (vii) SN addition. A SN will have three roles, in terms of data transmission, during the communication phase: (i) Role as a sensor, (ii) as a message forwarder, and (iii) as a broadcaster. In the following sections, we will use SN_i as a source node and SN_j as a neighbor to the source.

4.3.1. Transmission as a sensor

When SN_i senses data, it needs to send a message hop-by-hop to the BS. The SN_i only recognizes itself by its (PID_i), and the BS will recognize the source of the message by its PID_i as well. Thus, the PID_i of the source needs to be included in the message until it is received by the BS. Consequently, the PID of a sensor will be updated after every transmission. The SN_i needs to select one neighbor from N_i to forward the message to. The selection process goes through a probabilistic protocol which guarantees that SN_i does not use one neighbor all the time when forwarding its data; first, for routing privacy, and second for increasing the lifetime of the WSN. SN_i will form the message in the following format:

$$M_{i\rightarrow j} = OHPID_{i\leftrightarrow j} \,||\, E_{ki\rightarrow j}\,(PID_i \,||\, E_{ki\leftrightarrow bs}\,(D_i)) \qquad (4.9)$$

Where D_i includes the sensed data. Once SN_i knows that the message ($M_{i\rightarrow j}$) is delivered to the the neighbor, it needs to dispose the current pseudonym PID_i and issue a new one for the next transmission as indicated in expression (4.10):

$$PID_i = H_1(PID_i \oplus a_i) \qquad (4.10)$$

In addition, both SN_i and SN_j will dispose the current $OHPID_{i\rightarrow j}$ and issue a new one for the next communication between the two neighbors according to expression (4.11):

$$OHPID_{i\rightarrow j} = H_1(OHPID_{i\rightarrow j} \oplus a_{i\rightarrow j}) \qquad (4.11)$$

The message (M) will then be reformatted by the recipient SN_j and again forwarded to the next node, say SN_r, and so on, until it gets to the BS. If SN_j was the BS, then the BS uses the shared one-hop key between the sensor and the BS, to decrypt the data and to get the PID_i which the BS can use to recognize the source SN_i. Only at this point of time, BS can update the value of PID_i of SN_i. It also reads the data (D_i) which the BS can decrypt using $k_{i\leftrightarrow bs}$.

4.3.2. Transmission as a forwarder

When SN_i sends the message one-hop uplink to the neighbor SN_j, then SN_j needs to forward the message to another intermediary node. Upon receiving $M_{i\rightarrow j}$, SN_j will match $OHPID_{i\rightarrow j}$ in its table, T_j. If there is no match, then the message definitely is not addressed for SN_j and it will be dropped immediately. If it matches, then the message is decrypted using $k_{i\rightarrow j}$. The message will be forwarded to SN_r after (M) is reformatted as in (4.12):

$$M_{j\rightarrow r} = OHPID_{j\leftrightarrow r} \,||\, E_{kj\rightarrow r}\,(PID_i \,||\, E_{ki\leftrightarrow bs}\,(D_i)) \qquad (4.12)$$

Right after the data is *received* by SN_j and *forwarded* to the next one-hop SN_r, the SN_j updates the pseudonym $OHPID_{i\leftrightarrow j}$. SN_j now is ready to exchange another message with SN_i using the new pseudonym $OHPID_{i\leftrightarrow j}$. However, SN_j is not yet ready to send data

to SN_r since SN_r does not update the $OHPID_{j\leftrightarrow r}$ until (D_i) is forwarded to the next hop, say NS_v. See Figure 4.1 for the sequence of transmissions for a message from SN_i to the BS.

4.3.3. Acknowledgement

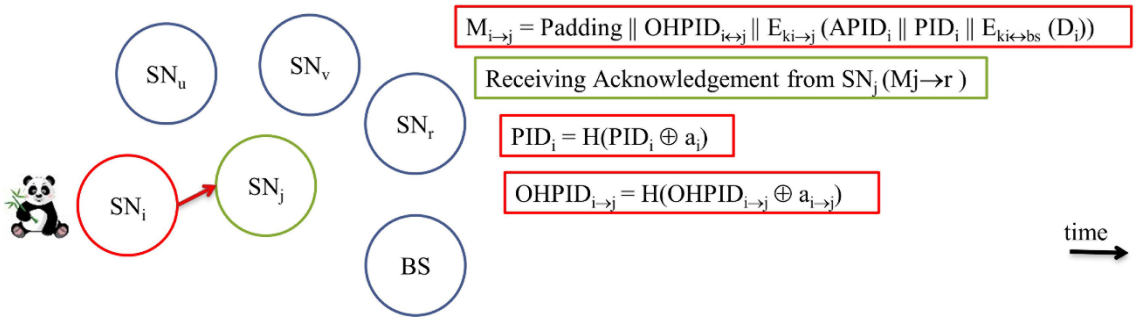
As expected in data networks, message could be lost or could get corrupted. In either case, retransmission is required. Because SNs change PIDs after each transmission, synchronizing PIDs is crucial. Updating the pseudonyms depends on successful message delivery. Ideally, the source should update the pseudonyms only after making sure the data is received by the BS.

However, the lack of direct connection between the source and the BS makes it a bit complicated process. The BS cannot send direct acknowledgement to the source if it is multiple hops away. We have to depend on multiple acknowledgements along the path between the source and the BS. SN_i needs to calculate the acknowledgement pseudonym ID ($APID_i$) according to expression (4.13):

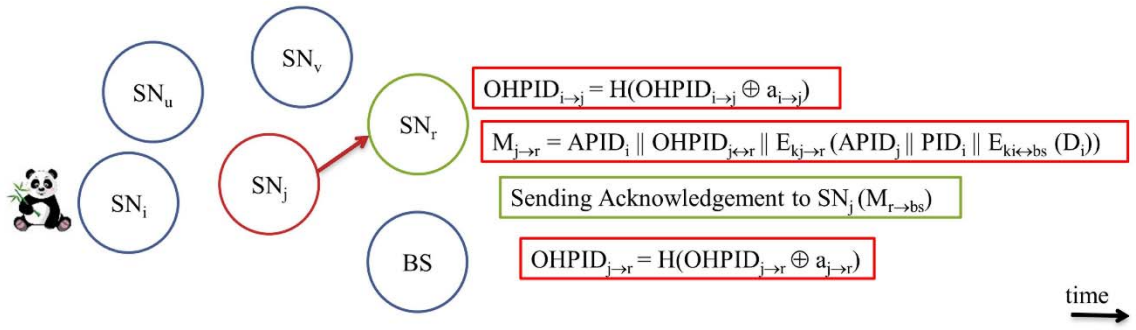
$$APID_i = H1(APID_i \oplus b_i) \qquad (4.13)$$

The message will be sent out to the neighbor with the current value for $APID_i$. Thus, we will rewrite $M_{i\rightarrow j}$ as it appears in (4.14):

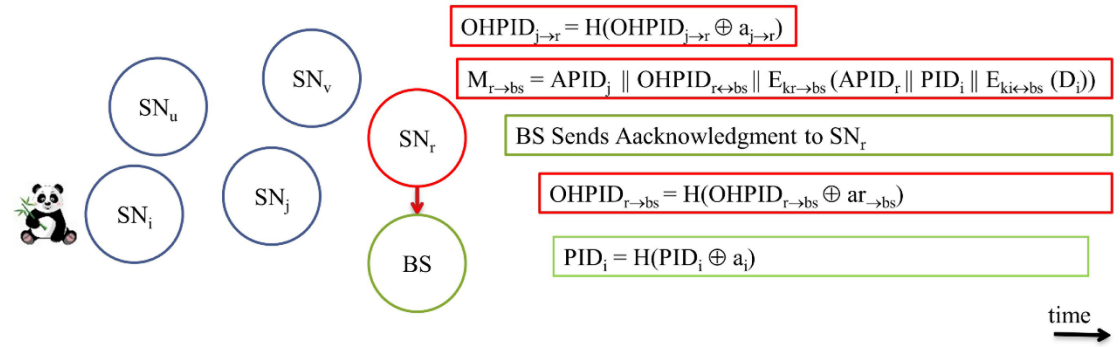
$$M_{i\rightarrow j} = \textbf{Padding} \,||\, OHPID_{i\leftrightarrow j} \,||\, E_{ki\rightarrow j}(\textbf{APID}_\textbf{i} \,||\, PID_i \,||\, E_{ki\leftrightarrow bs}(D_i)) \qquad (4.14)$$



(a) SN_j receives a message from SN_i.



(b) SN_j forwards the message to a neighbor SN_r.



(c) BS receives the message and processes it.

Figure 4.1: The sequence of a message transmission from SN_i to the BS.

Padding is added to make sure all the one-hop messages have the same size to prevent *size correlation* attacks. When SN_j receives the message, it will reformat the message as in expression (4.15) and then send it to SN_r:

$$M_{j\rightarrow r} = \mathbf{APID_i} \;||\; OHPID_{j\leftrightarrow r} \;||\; E_{j\rightarrow r}(\mathbf{APID_j} \;||\; PID_i \;||\; E_{ki\leftrightarrow bs}(D_i)) \qquad (4.15)$$

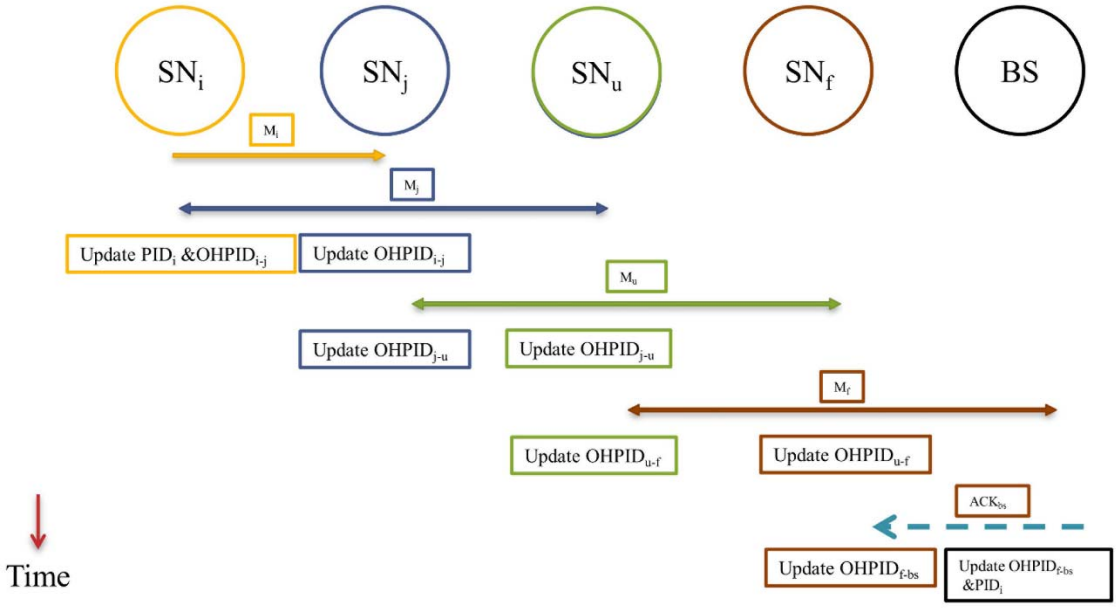


Figure 4.2: Using $APID_i$ for acknowledgement with no errors.

The transmission of $M_{j\rightarrow r}$ should be heard by all the neighbors including both SN_i and SN_r. If SN_i hears the message and reads ($APID_i$), the SN_i knows that $M_{i\rightarrow j}$ was received correctly by SN_j. Only at this time SN_i updates the value of $OHPID_{i\leftrightarrow j}$. PID_i will get updated, as well, since SN_i is the source of the message. Here are two scenarios:

Scenario 1: The packet sent by SN_i is lost or got corrupted. In this case, SN_j considers nothing happened, so it will not forward any message onward. Meanwhile, SN_i will wait for *(ζ)* time to expire. It will send the message again with updated $APID_i$. Once the message gets acknowledged according to the procedure explained earlier, then PID_i, $OHPID_i$ and $APID_i$ will get updated. If it is intermediary SN, only $OHPID_i$ and $APID_i$ gets updated as exhibited in Figure 4.3.

Scenario 2: The packet is received correctly by SN_j, the new packet $M_{j\rightarrow r}$ is sent out which contains the acknowledgement ($APID_i$), and SN_j updated the value of $OHPID_{i\leftrightarrow j}$. However, SN_i does not hear the forwarded message $M_{j\rightarrow r}$ within time *(ζ)*. At this moment SN_i does not know for sure if the message was delivered *(resembles scenario 1)*, or the acknowledgement is lost. It has to account for the worst case. A copy of the message will be retransmitted to SN_j with the current $OHPID_i$ and updated $APID_i$. SN_j can recognize the message because of the value of old $OHPID_i$. After receiving the *retransmitted* message, it now sends a direct acknowledgement to SN_i as in expression (4.16).

$$ACK_{i\leftarrow j} = APID_i \,||\, \text{Padding} \qquad (4.16)$$

Figure 4.4 exhibits the process. BS is treated similar to a normal SN, so it has to acknowledge every message it receives. After the message is delivered to the BS, and after the message is acknowledged, the PID_i (of the source) will get updated on the BS tables while it has been already updated in the sensor itself after the first acknowledgement.

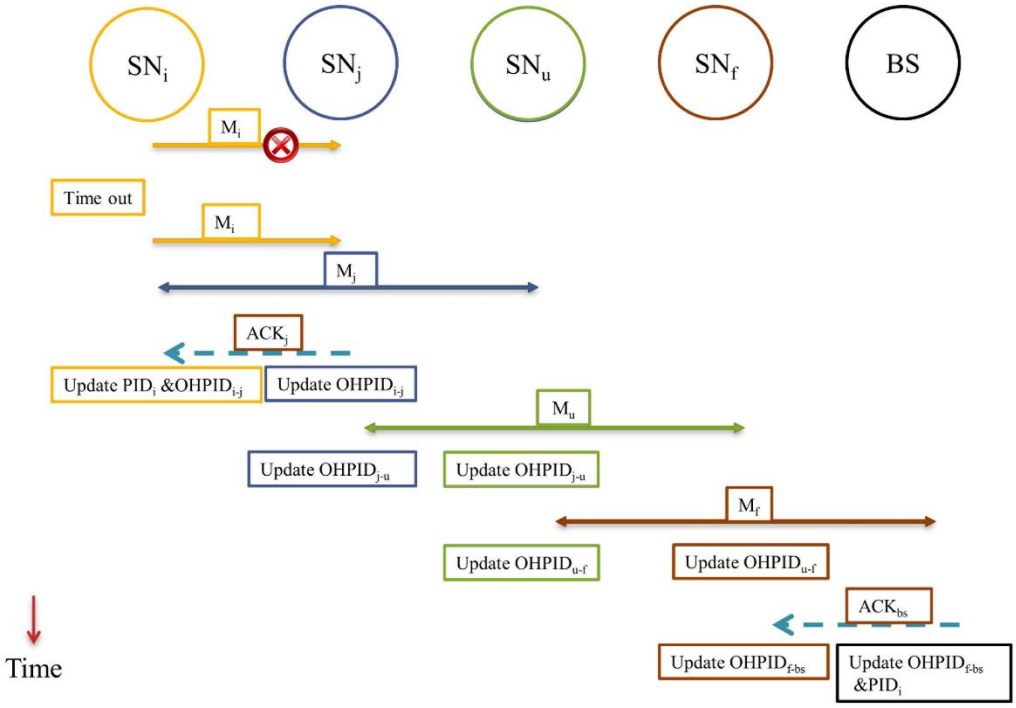


Figure 4.3: Acknowledgement for a message with errors.

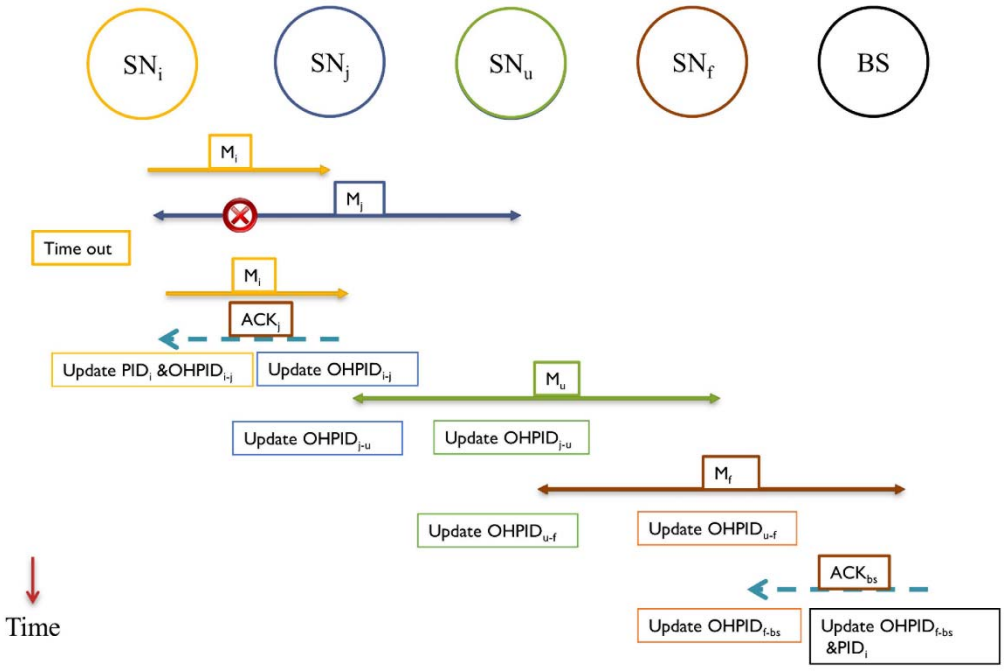


Figure 4.4: Handling lost acknowledgement.

Both the SN_i and the BS will be ready to exchange a new message. As long the new message does not reach to the BS before the old PID_i gets updated, the system will remain synchronized. This way, we have a possible window of one message only. We propose implementing a *sliding window* mechanism as exhibited in Figure 4.5 [7]. For each sensor, we can have a window of *(W)* slots.

4.3.4. Transmission as a broadcaster

Typically, the BS is required to broadcast a message for control and management purposes. Likewise, a sensor might need to broadcast a message to the BS or to the neighbors for network setup, maintenance and other management issues. The framework requires keeping all the messages indistinguishable throughout the network, so all the messages need to have the same size. Each SN is preloaded with a broadcast key (kb_i) and assigned broadcast pseudonym ($BPID_i$). The broadcast message sent by SN_i is formatted as in expression (4.17):

$$M_b = \text{Padding} \,||\, BPID_i \,||\, E_{kbi}(D_b) \qquad (4.17)$$

The broadcast message from a source SN_i will be received by all the neighbors. SN_i and the recipients will update $BPID_i$ according to expression (4.18).

$$BPID_i = H_1(BPID_i \oplus b_i) \qquad (4.18)$$

Upon receiving the broadcast message (M_b), SN_j decrypts the message using (kb_i) stored in the table (T_j). It then encrypts it again using (kb_j) and broadcasts (M_b) to its one-hop neighbors set (N_j) as in expression (19):

$$M_b = BPID_i \,||\, BPID_j \,||\, E_{kbj}(D_b) \qquad (4.19)$$

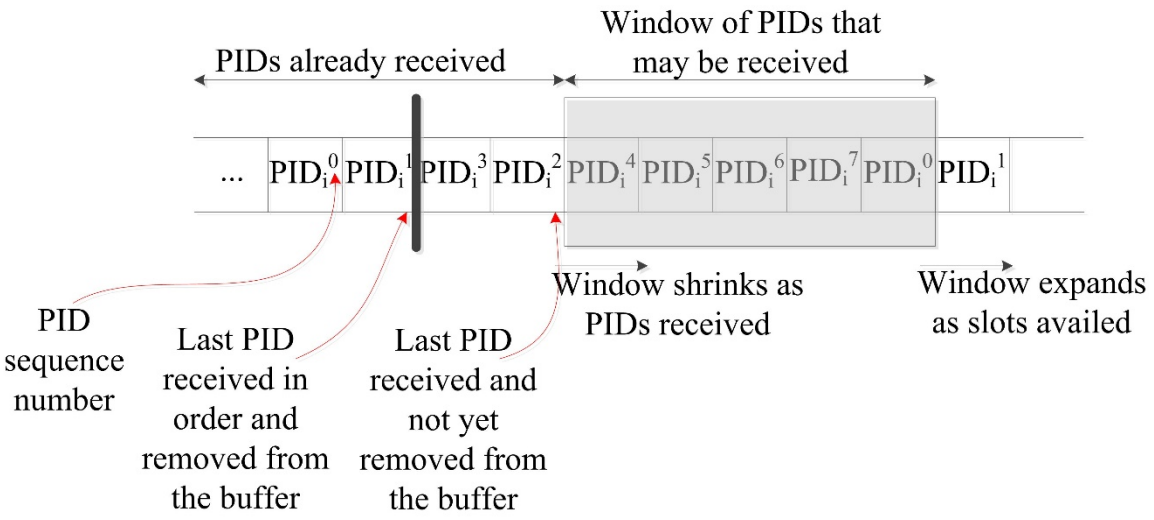


Figure 4.5: Sliding window for received PIDs [7].

When the BS receives a broadcast message, it is ultimately the destination, so intuitively it does not need to broadcast the message again. Our proposed framework assumes that the BS behaves similar to a normal sensor. To maintain this pre-course, we require the BS to broadcast the message again for acknowledgement purpose. Thus, we introduce the limited broadcast where the BS will be able to broadcast to only one hop (TTL=1).

4.3.5. Limited Broadcast Messages

A sensor inside network maze can only recognize the neighboring sensors and the BS. When SN broadcasts a message uplink (towards the BS), then all neighbors should hear it. The neighbor should broadcast the message again if and only if the message comes from a SN with a bigger hop-count (HC). This will conserve a lot of unnecessary traffic and energy dissipation. The broadcast message will contain (*TTL=HC)*. The value will

keep decreasing by one until it gets to the BS. In contrast, the downlink broadcast messages (by the BS to the SNs) should have (TTL=0) where the intermediary sensors would rebroadcast the message if and only if it comes from a neighbor with a smaller (HC). A special case when (TTL=1) where the message will be broadcasted to one-hop neighbors only. FAC also may adopted a more sophisticated optimized flooding algorithms for wireless multi-hop network, such as CDS-based algorithms [81, 82].

4.3.6. Fake Broadcast Message

The sensors need to send fake messages to prevent *time correlation, rate analysis* and *statistical analysis*. A fake message is technically a one-hop broadcast message. However, to prevent correlation, the message needs to behave similar to real messages. So, the message needs to be encrypted and have similar size as the real message to make it completely *indistinguishable*. Since it has to carry a dummy data, it will contain the *residual energy (Δ)* of the issuing sensor. This information will be extracted by the recipient neighbors and saved in the related tuple in the table (T). The fake broadcast message sent by SN_i is as in expression (4.20):

$$M_f = \text{Padding} \,||\, FPID_i \,||\, E_{kfi}(\Delta_i) \qquad (4.20)$$

The fake broadcast message from SN_i will be received by all the neighbors. SN_i and the recipients will then update $FPID_i$ according to expression (4.21).

$$FPID_i = H_1(FPID_i \oplus c_i) \qquad (4.21)$$

There is no need to worry about the pseudonyms synchronization since the main purpose of the fake messages is to show activity in idle sensors to obfuscate real messages.

4.4. SN removal

There are many reasons why a sensor should be removed from WSN. For instance, when the battery of a sensor is about to deplete, it should refrain from participation. This would protect against data loss and maintain the pseudonyms synchronized. In some other cases, WSN use IDS [83, 84] to protect against active attacks, so once a sensor is captured, it must be banished from the network.

Procedurally, if SN_i opts to be removed, it will send a messages to the BS as in expression (4.22).

$$M_{i\rightarrow j} = OHPID_{i\leftrightarrow j} \,||\, E_{ki\rightarrow j}\,(PID_i \,||\, E_{ki\leftrightarrow bs}\,(D_{remove})) \qquad (4.22)$$

Where (D_{remove}) is a command to banish the sensor. The tuple of the SN_i in the BS tables will be disabled permanently. In addition, SN_i will send a broadcast message to the neighbors as in (4.23):

$$M_b = Padding \,||\, BPID_i \,||\, E_{kbi}(D_{remove}) \qquad (4.23)$$

Once the neighbors get the message (D_{remove}), they will delete the tuple related to SN_i from the table (T) and banish the sensor. The same process could be used by the BS for sensor removal.

4.5. SN Addition

To add a new sensor to the network, the sensor will be preloaded with the required parameters: ID_i, a_i, b_i, c_i, H1, $k_{i\leftrightarrow bs}$ and kb_i, and fkb_i. Right after deployment, the sensor calculates the shared parameters with its neighbors. The BS should be trusted to run the process. The BS will send special key *(k_{add})* to all the neighbors. SN_i will be preloaded with the same key as well. SN_i and the neighbors will use this special key to authenticate with each other. Initially, the BS sends the following message to the one-hop neighbors of the new sensor as in expression (4.24):

$$M_b = \text{Padding} \,||\, BPID_{bs} \,||\, E_{kb\text{-}bs}(D_{add}) \qquad (4.24)$$

Where (D_{add}) is expressed in expression in (4.25):

$$D_{add} = hc \,||\, k_{add} \qquad (4.25)$$

The initial value for *hc* is *zero*. It will be incremented every time the message is forwarded.

CHAPTER 5: DATA AUTHENTICATION AND INTEGRITY

The data is encrypted before transmission to protect against passive attacks such as eavesdropping. For active attacks, such as data and transaction falsification, message authentication is required. The two important security aspects to achieve: (i) Verify that the content of the message is not altered and, (ii) the source is authentic. We could achieve authentication by either using a message authentication code *(MAC),* or one way hash function *(OWH).* MAC would require the sender (SN_i) and receiver (BS) to share a secret key. The authentication code is calculated as: *MAC = F(k, D)*. *DES* or other algorithms can be used to generate the code. The OWH also accepts a variable size message *(D)* as input and produces a fixed sized digest *MD = H(D)* as output. Examples for OWH are: *SHA, MD5, Whirlpool and HMAC.* The advantage of OWH over MAC is the fact that it does not use encryption which is quite slow. Comes in the middle, HMAC which is a MAC derived from OWH such as *SHA-1*. It could be expressed as: *MD = HMAC(K, D)*.

If we opt to use HMAC as an example, the ($M_{i\rightarrow j}$) will be rewritten as in expression (5.1):

$$M_{i\rightarrow j} = APID_i \,||\, OHPID_{i\leftrightarrow j} \,||\, E_{ki\rightarrow j}\,(APID_i \,||\, PID_i \,||\, E_{ki\leftrightarrow bs}\,(D_i)) \,||\, \mathbf{HMAC_{ki\leftrightarrow bs}\,(PID_i \,||\, D_i)} \quad (5.1)$$

The key ($k_{i\leftrightarrow bs}$) is shared between SN_i and the BS. The message could be authenticated with MD using OWH as in the expression below:

$$M_{i\rightarrow j} = APID_i \;||\; OHPID_{i\leftrightarrow j} \;||\; E_{ki\rightarrow j}\,(APID_i \;||\; PID_i \;||\; E_{ki\leftrightarrow bs}\,(D_i) \;||\; \mathbf{H(PID_i \;||\; E_{ki\leftrightarrow bs}\,(D_i))})$$

(5.2)

As it is transparent from expression (5.2), we need more processing time and therefore more power consumption because we have encrypted a sizable packet. There is a tradeoff between higher security and energy conservation. The first approach is more appropriate. Authentication for the broadcast messages is done as in expression (5.3):

$$M_b = Padding \;||\; BPID_i \;||\; E_{kbi}(\,D_b) \;||\; HMAC_{kbi}(\,D_b)\,) \qquad (5.3)$$

Alternatively, it can be achieved using expression (5.4):

$$M_b = Padding \;||\; BPID_i \;||\; E_{kbi}(\,D_b \;||\; H(kb_i \;||\; D_b)\,) \qquad (5.4)$$

The message could contain other important information such as *sequence number* (similar to the well-known *HDLC* and *TCP* protocols) and *time stamp*. The receiver uses the sequence number to verify the order of messages. Time stamp is used to check the delay threshold. Both checks will enhance protection against various active attacks. The message core data (D_i) could have the following format:

$$D_i = \; SEQ\_NO \;||\; TIME\_STAMP \;||\; MSG\_LGTH \;||\; SENSED\_DATA \qquad (5.5)$$

CHAPTER 6: TEMPORAL AND RATE PRIVACY

WSN could suffer from time correlation attacks [6, 18, 19, 28, 85] by observing the time between correlative packets sent and received in a certain neighborhood. The adversary can trace forward and backward the messages until they reach to the BS or to the source. Hence, hiding temporal information is crucial for both anonymity and location privacy. Using routing schemes to protect against time correlation attacks is found to be efficient to certain extent where local adversary usually has limited mobility and partial view of the network traffic. However, routing based schemes do not work for global adversary where the traffic of the whole network can be easily monitored with a full spatial view and the adversaries can collude together to promptly detect the origin and time information of the event [34, 70]. A mechanism is required to divert attention of the adversary when there is event-driven transmissions, especially with the presence of global adversary [36]. The distribution of events changes which could be a reason for the adversary to detect the event timing and thereafter the source of the event. The message distribution (both real and fake) needs to be adjusted to prevent time correlation. In some applications, such as monitoring and surveillance, we cannot guarantee a certain event distribution. The literature talk about three ways to maintain an obfuscated message distribution: (i) By issuing message delays, and (ii) by issuing fake messages, and (ii) by using both delays and fake messages. Using delays works well against local adversary but might not be suitable for time sensitive networks. In contrast, using fake messages is required to protect against multi-local and global adversary, however, it is very expensive

in terms of energy dissipation. Furthermore, adversary with good statistical analysis can easily detect the message distribution if the scheme is not designed carefully [70].

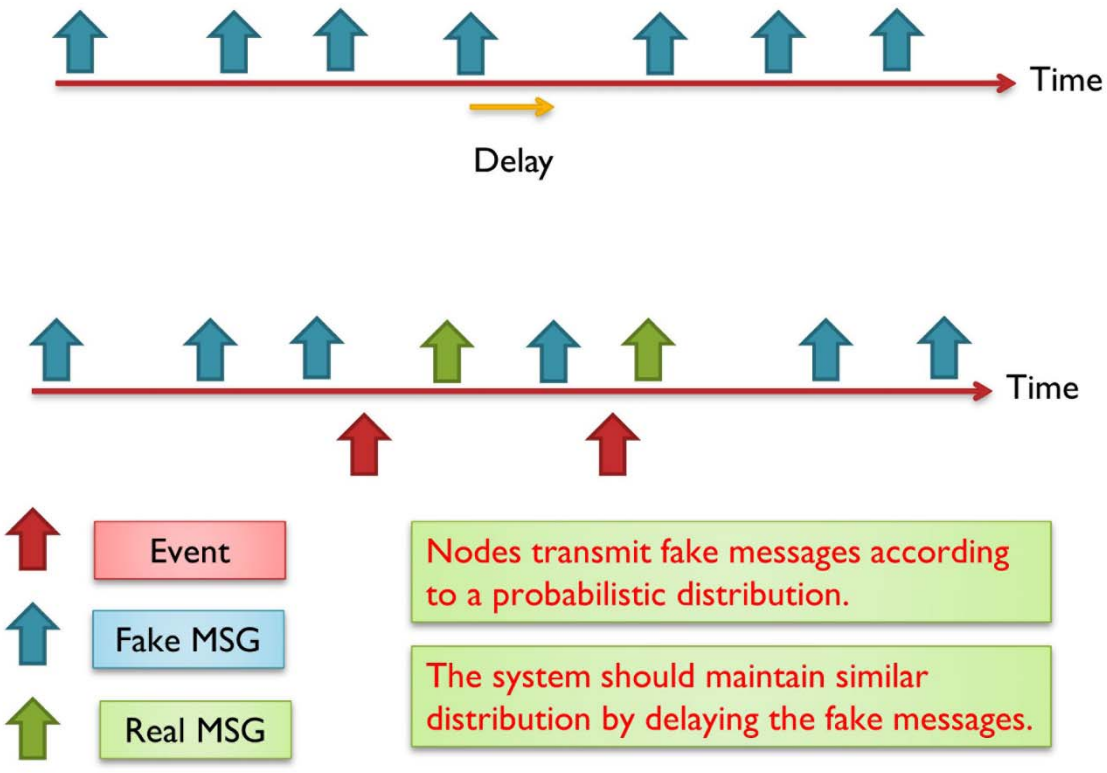


Figure 6.1: A probabilistic distribution for fake messages [41, 70].

Some work in the literature clearly differentiates between two terms: the *event* (of transmission) and the *interval* (of transmission). If every interval has only one transmission, then event and interval are the same, however, this might not be the case when we have multiple transmissions during one interval. So, the anonymity level depends on the capability of the adversary to distinguish between real and fake transmissions. This means, given multiple transmissions by a SN, the adversary must be unable to distinguish, with significant confidence, between transmissions carry real data and transmissions carry fake data. Alomair et al. [70] suggested that transmission "indistinguishability" is not

enough. They claim that indistinguishability is achieved when adversary monitoring the network over multiple time intervals, in which some intervals contain real event transmissions and others do not, is unable to determine, with significant confidence, which of the intervals contain the real traffic. If intervals are indistinguishable, the individual transmissions within the interval should also be indistinguishable.

We should have a mechanism to quantify anonymity while it is used, in the literature, in different ways. However, in our work, anonymity means how to prevent the adversary from knowing the source of the message. In other words, the adversary could know that a particular sensor sent a message at one time, but it should not know that sensor is the source of the message. By delaying the real messages and by issuing multiple messages at one interval would mislead the adversary. As an example, for one transmission and one adversary, where the adversary can guess either the message is real or fake without any anonymity measurement taken, it should be 0.5 (either fake or real). Let's presume ψ donates one adversary strategy for breaching the anonymity of the system among a set of strategies. Let's presume P_r is the probability that the adversary succeeds using strategy ψ. The anonymity A as defined in [70] with the existence of a strategy ψ, is presented in the expression below:

$$A_{\psi} := 1 - P_r, \text{ where } \quad 0 \leq P_r \leq 1 \qquad (6.1)$$

If we presume that Σ represents all possible strategies for the adversary to breach the anonymity of the WSN, the accumulated anonymity will be as in the expression below:

$$A := \min(A_{\psi}), \text{ where } \psi \in \Sigma \qquad (6.2)$$

It is very important to increase anonymity for every individual SN in the network especially with the presence of multi-local or global adversaries. Presence of colluding adversaries could cause the anonymity to drop exponentially [70]. Take Figure 6.2 as an example, where WSN has a moving Panda from point "a", to "b", to "c", then finally to "d" where each location has a SN to report the Panda's movement. If the anonymity of each sensor is $A=0.8$, then the anonymity at node "b" is $A=0.8^2=0.64$ and at point "d" is $A=0.8^4=0.41$. Having global adversary makes it super necessary to design a strong anonymity model which can resist the time correlation attack [6].

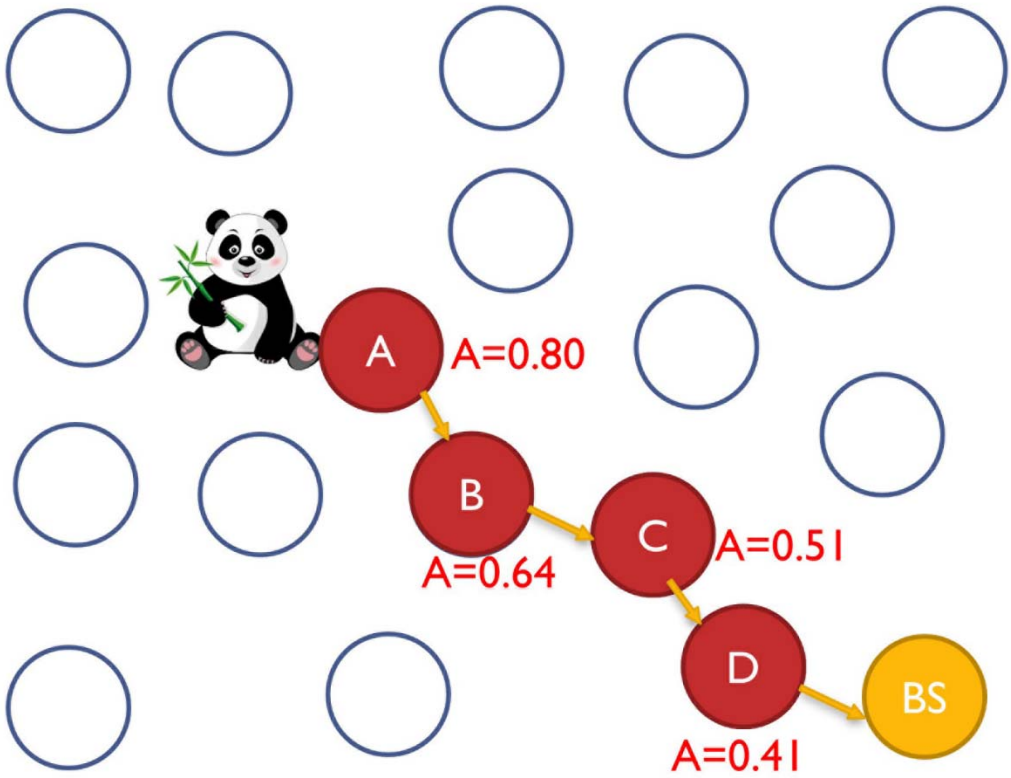


Figure 6.2: Having multiple colluding nodes will reduce system anonymity exponentially [6, 41, 70].

In this work, we assume the worst case for time correlation attacks which is a global or laptop-class adversary attacks [10]. Having an anonymity scheme to protect against the

global adversary will be very expensive solution in terms of energy preservation and thus the lifetime of the network. In the following two subsections, we propose two schemes, the simple global anti temporal *(SGAT)* and the energy controlled anti temporal *(ECAT)*.

6.1. Simple Global Anti Temporal scheme (SGAT)

When an event-driven message is sent out, the adversary can trace back the message to the SN or forward to the BS. Sending few other transmissions in the network within the range of the adversary confuses it and prevents the adversary from having known path to follow.

In this work, we presume the lifetime of the network *(Ω)* is divided into a number of intervals *(I)* and each interval time is *(ω)*, where:

$$\Omega = I \,.\, \omega_i \qquad (6.3)$$

The value of Ω can be predicted as a range between a minimum value (worst case) Ω_{min} and a maximum value (best case) Ω_{max}. It all depends on how real / fake transmissions are facilitated. The SNs will send either a fake or a real message during one interval. The message is sent at the end of each interval or it is adjusted to be sent during the interval to create some variable delays through the route to the BS which would confuse the adversary more and would prevent it from gaining useful knowledge about the network based on time correlation. SN_i which has sensed the event or received the real data from another SN_j, will send the real message *(M_r)* through a hop-by-hop path to the BS, and some other nodes will

send fake messages *(M_f)* during the same time to disrupt the adversary. There are two questions:

How long the message will be held in the SN after it is sensed? Simplistically, M_r and M_f are sent at the end of the interval *I*. The time period from arrival of the data to the end of the period time (τ_w) expressed in the expression below:

$$\tau_w = \omega_i - t_a \qquad \text{where: } t_0 \leq t_a \leq t_s \leq \omega_i \qquad (6.4)$$

Where: t_0 is the beginning of the interval I_i, t_a is the arrival time, $t_0 \leq t_a$.

Ideally, the message will be sent immediately after it is sensed or received which makes $\tau_w = 0$. Theoretically, τ_w could be a value: $0 \leq \tau_w \leq \omega_i$ as exhibited in Figure 6.3.

How many SNs in the network will send fake message during one interval time and which ones? Simplistically, every SN in the network which is in the range of the adversary and in the range of source SN, should send a fake message while SNs that have real messages will send the real messages only.

There are many technical issues regarding the determination of the optimal configuration for both questions mentioned earlier. For instance, it is not possible for the neighboring nodes to know in advance when a SN is going to sense an event. It is a completely unpredictable random-distribution for the events. The need to transmit fake messages becomes even much more crucial if we do not have busy-network. Therefore, all SNs with no real messages need to send fake messages during the interval I_i, in the worst case, or only a selected nodes according to a probabilistic protocol. Having high number

of fake message transmissions will reduce the lifetime of the network in favor of privacy. Doing the reverse will jeopardize the privacy of the sensor nodes. Sending real and fake messages at the end of the interval could be learned by the adversary. However, it is not very dangerous if the network sends enough fake messages at the same time.

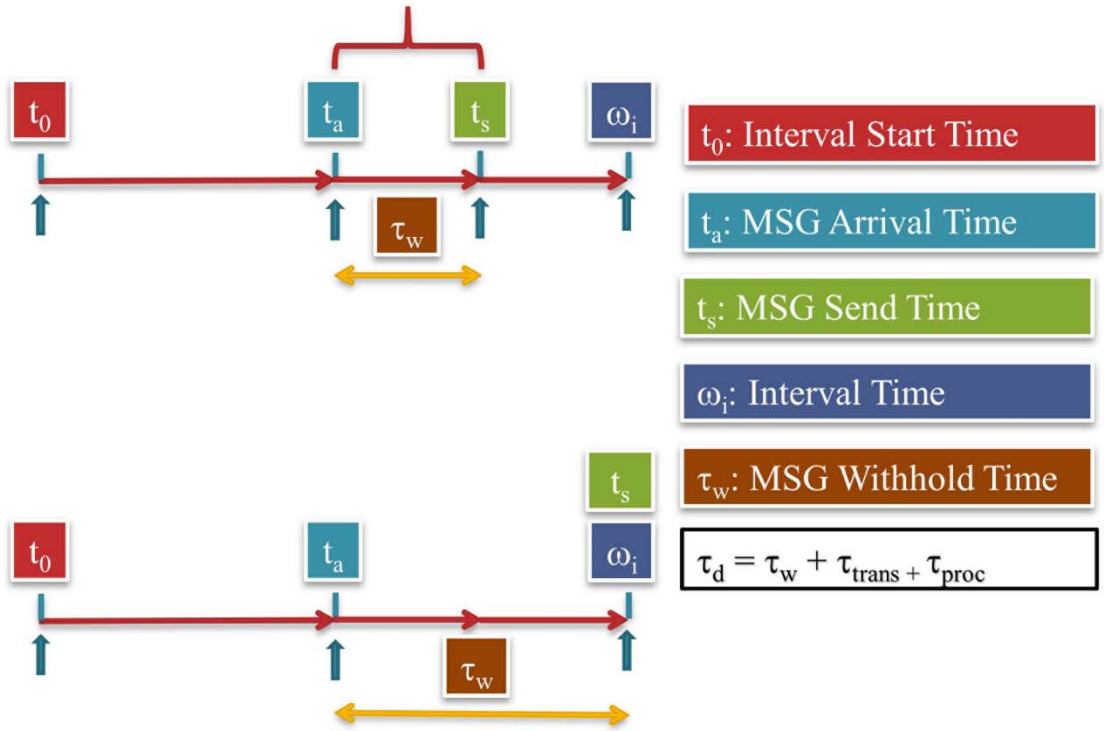


Figure 6.3: Delays incurred at each SN [6].

Having variable withhold time (τ_w) is useful for privacy and for reducing the average network delay. The delivery time (τ_d) presuming that the message is always sent at the end of the interval I_i is:

$$\tau_d = \tau_w + \tau_{trans} + \tau_{proc} \qquad (6.5)$$

Where: τ_d is delivery time, τ_{trans} is transmission time, τ_{proc} is processing time.

If we presume τ_{proc} is much smaller than τ_{trans}, then τ_d can be rewritten as the following:

$$\tau_d = \tau_w + \tau_{trans} \qquad (6.6)$$

If the message needs to go through *(U)* hops to the BS, and if we assume that the transmission only happens at the end of the Interval I_i, then t_s, equals to ω_i, and the total delivery time $(\tau d\text{-}_{total})$ can be calculated according to the expression below [6]:

$$\tau_{d-total} = \sum_{u=1}^{U} \tau_{w_u} + \tau_{trans_u} \qquad (6.7)$$

Having t_s equals to ω_i; i.e., sending message at the end of the interval, will increase the delay of the delivery presuming that every τ_{trans} is equal [6]. Thus, optimizing $\tau_{d\text{-}total}$ is a function of τ_w according to the expression :

$$\tau_{d-total} = f(\tau_w) = \sum_{u=1}^{U} \tau_{w_u} \qquad (6.8)$$

Each SN will be informed during the setup phase about ω_i for the lifetime of the network. The BS also can alter this value by broadcast when the conditions of the WSN changes (closed-loop control). The value of ω_i should be calculated to achieve at least the minimum expected lifetime span Ω_{min} without jeopardizing the privacy and data integrity. Thus,

$$\Omega_{high\text{-}th} \geq \Omega_i \geq \Omega_{low\text{-}th} \qquad (6.9)$$

Where: $\Omega_{high\text{-}th}$ is the highest possible value for Ω_i and, $\Omega_{low\text{-}th}$ is the lowest expected value for Ω_i. When SN does not have a real message to send before the end of the interval period I_i, it will send a fake message according to the procedure explained in the anonymity module. When SN has a real message, it will send it to one node from the neighborhood set (N_i).

6.1.1. Security analysis

The adversary sees every SN sending a message at a fixed data rate at any one time. It also cannot distinguish any message from the rest of the messages in the network since none have similar ID. If we have N nodes in the WSN, the probability that one adversary can locate the sending node equals to:

$$P_r = \frac{1}{N} \qquad (6.10)$$

We can calculate anonymity as:

$$\mathrm{A} = 1 - P_r \qquad (6.11)$$

6.1.2. Delivery time

Message follows hop-by-hop path until it gets to the BS as exhibited in Figure 6.4. In this scheme, the message waits until the end of the interval. The delay will be calculated according to the expression below [6]:

$$\tau_{d-total} = \tau_{w_0} + (\mathrm{HC} - 1) * \omega_i + \tau_{trans_u} \qquad (6.12)$$

It axiomatic that most delay accumulates from holding the message until the end of the interval periods.

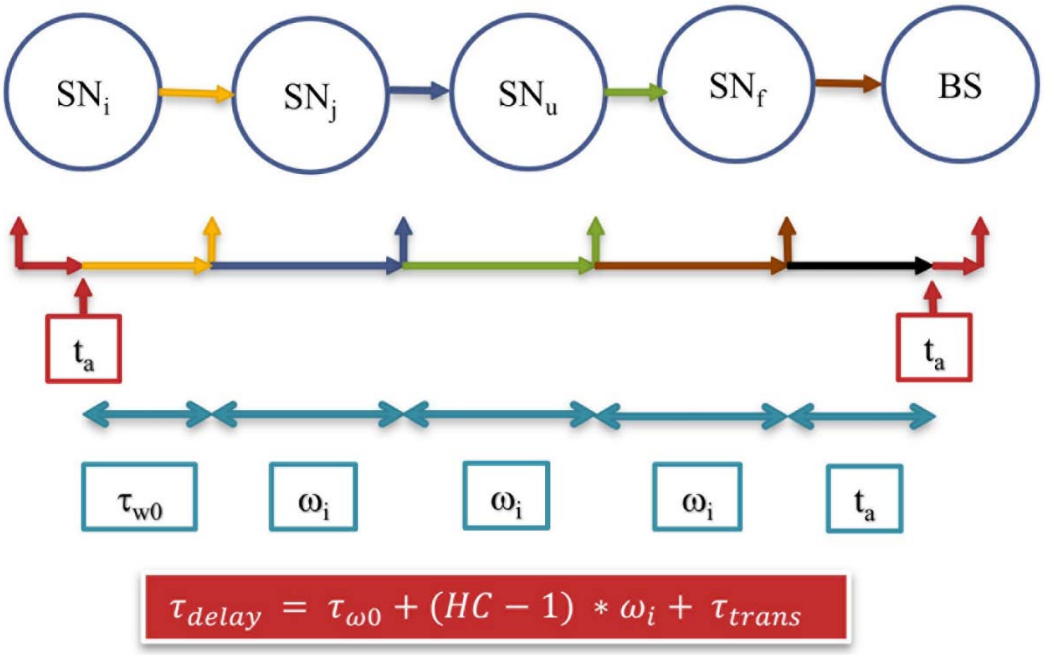


Figure 6.4: Total delay required to send a message from a source to the BS through (U) hopes [6]

6.1.3. Energy cost

We presume in this scheme that every node would send one message at the end of each interval. The message could be either real or fake. If we have *(N)* nodes in the WSN, then we expect *(N)* messages during each interval I_i. The energy spent for transmission is almost constant since we have fixed size messages. However, we can evaluate how expensive it would be to use fake messages for privacy enhancement. If we have *(Q)* percent of the nodes send real messages at each interval, then we are wasting *(1-Q)* percent

of the energy and of the bandwidth. We can adjust the amount of energy consumed by increasing the interval period ω_i. However, increasing ω_i, would increase the delays.

If a SN receives multiple messages in one interval, then it will *queue* the messages for transmission. Because the SN needs to wait till the end of the interval, it could arrange the messages in a queue and send them randomly at the end of interval. This should also increase the privacy and security of the data. It could also select a different forward node for each message. In conclusion, SGAT is energy-expensive due to sending fake/real messages by every node per each interval of time. However, SGAT provides the maximum message *entropy*. Figure 6.5 exhibits the network transmissions for two consecutive intervals.

6.2. Energy Controlled Anti Temporal scheme (ECAT)

There are three major drawbacks in *SGAT*: (i) Having fixed interval time ω_i while it is possible to adjust the value for a better traffic and energy control, (ii) not considering the residual energy as a metric for selecting the forward hop, (iii) high rate of traffic due to fake messages.

6.2.1. Changing ω_i from fixed to variable

Having a fixed interval time ω_i could be a glitch for network performance. If ω_i is set to be a large value, then the delay will be high which could be a serious problem in some time sensitive applications. If ω_i is set to be a small value, then a huge amount of fake messages will be sent at the end of each interval which will reduce the lifetime of SNs

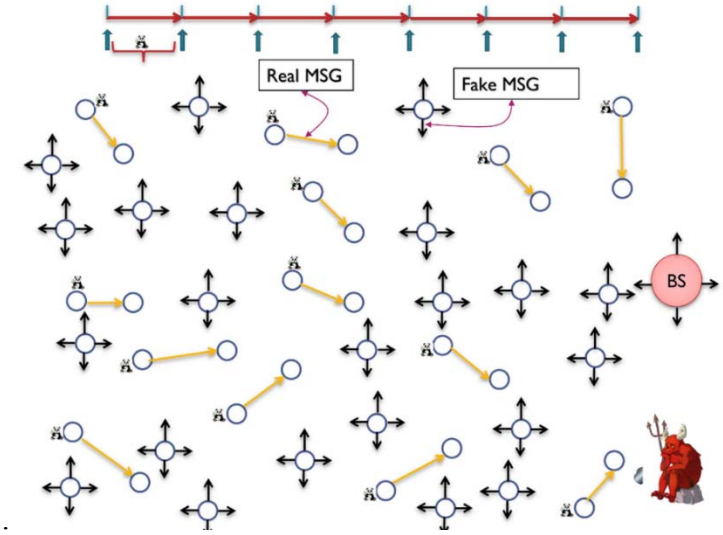
and accordingly the lifetime of the WSN. We propose that we have variable ω_i as presented in [10]. Every node will be calculating its ω_i using a pseudo-random number generator (PRNG). We suggest a uniform distribution algorithm such as multiplicative congruential algorithm [86, 87], which is the basis for many of the random number generators in use today. Lehmer's generators [88] involve three integer parameters, *r*, *s*, and *m*, and an initial value, x_0, called the seed. A sequence is generated by the following modified formula:

$$X_{k+1} = b*((r.X_k + s) \bmod m + f) \qquad (6.13)$$

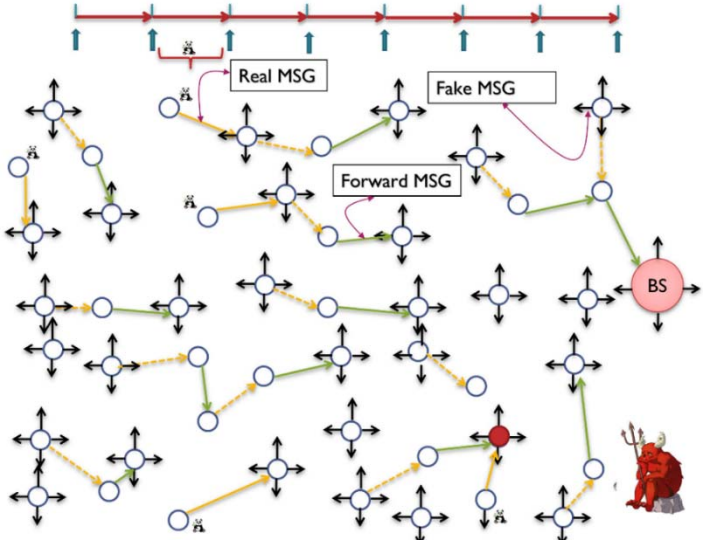
The result of the modified PRNG will be a sequence of integer values between (b×f) and (b×(m+f-1)). Each SN needs to be preloaded with the seed x0, r, s, m, b and f values. The seed range is 0 to (m-1) and it is uniformly assigned to the sensor nodes. If b = 2, f = 1 and m = 4 then sequence of four intervals will be: $\omega_i \in [2,4,6,8]$ time-units as exhibited in Figure 6.6. We could have up to (m!) different sequences which are uniformly distributed on the SNs. For instance, we can have (24) different sequences for our example and if we have (48) nodes in the network, so each sequence should be provided to two nodes only.

Each node will be dynamically assigned an interval value which needs to change after each transmission. Taking the example above, the first $\frac{1}{m}$ th (more or less) of the sensors will send data after *2 time-units.* Then, the second $\frac{1}{m}$ th will transmit after *4 time-units,* and so on. At any point of time, the adversary will be faced by enough transmissions in the network that it could divert its attention far away from the SNs sending real data. By having *(m)* interval values where each SN will be generating one ω_i using the PRNG, we

have reduced the average interval time from ω_{max} to ω_{ave}. That is explained in the inequality below:



(a) Sensors sense events and send real messages while the rest send fake messages.



(b): Sensors send or forward real messages will not send fake messages. The more the network gets busy the less fake messages are transmitted.

Figure 6.5: Anonymity with fake messages.

$$\omega_{max} > \omega_{ave} = \frac{T_1 + \cdots + T_m}{m} > \omega_{min} \qquad (6.14)$$

Considering the earlier example, we have $\omega_{max}=8$, $\omega_{min}=2$ and $\omega_{ave}=5$. That is: we have reduced the delay interval by 37.5%. If $m=8$, then delay reduced by 44%. The transmission of real and fake messages is exhibited in Figure 6.7.

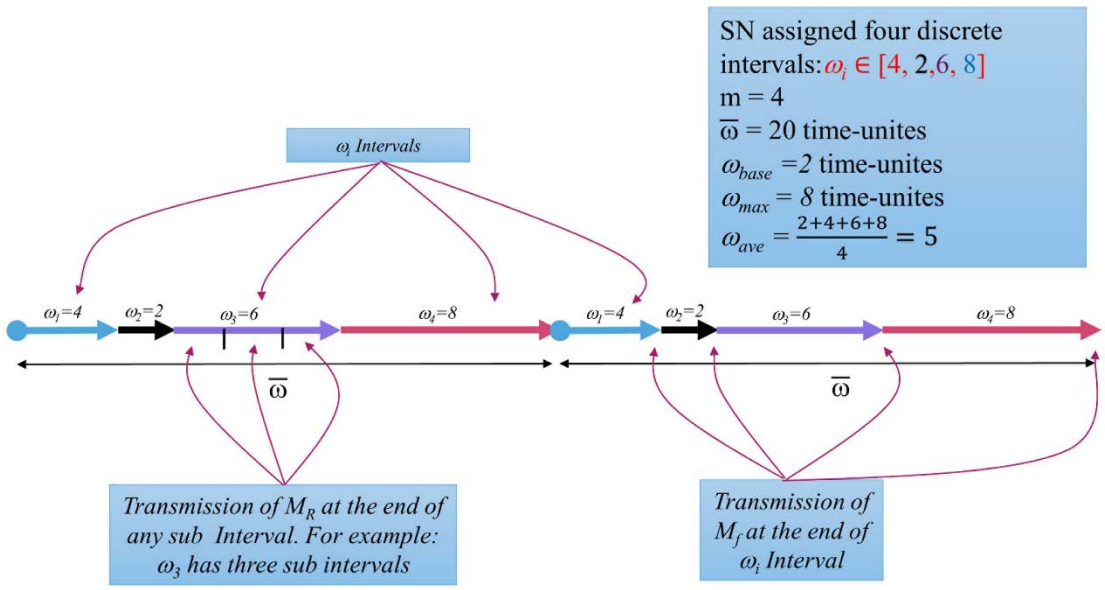


Figure 6.6: The generation of the sequence of intervals assigned for each sensor.

6.2.2. Reducing fake messages and delay for real messages

We have created a mechanism for dynamic interval allocation. Let us call $\overline{\omega}$ the *big interval* which has *subintervals* ω_i. It still makes sense to send fake messages at the end of each interval. However, having the real message wait until the end of the interval time, as exhibited in Figure 6.8, is not commendable because it increases the delay at each node. Let us presume the current subinterval ω_i is the maximum, which is *8 time-units* according

to the example discussed earlier. Let us presume that the message was sensed at *2 time-units* and it is ready to be sent at *4 time-units*. Following the SGAT rules, it still needs to wait for another *4 time-units* to be sent out! However, we know for sure that many other nodes have different ω_i subinterval values. Thus, during the time subintervals 2, 4, 6, and 8 there is enough traffic in the network. We propose that when the data is ready, the SN_i should send the data during the next subinterval slot within the current interval ω_i. Consequently, if we are at interval $\omega_{max}=8$ which has four subintervals at (2, 4, 6, and 8), and for our example, at *6 time-units* the data can be sent out. This way, we save about *2 time-units* delay while we can guarantee that the adversary will not be able to infer the source of transmission because we have enough traffic distributed in the network.

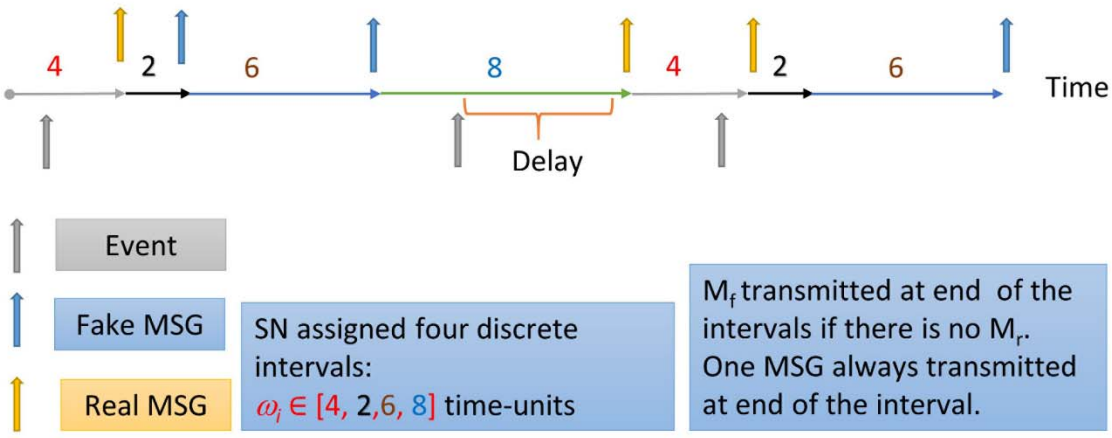


Figure 6.7: SN is assigned a sequence of intervals which repeat until sensor lifetime ends.

If we select higher values for $\overline{\omega}$, then we can further reduce the number of fake messages transmitting at one subinterval, however, we are increasing the average delay as well. Selecting a value for $\overline{\omega}$ could be a tool to adjust security versus energy conservation. We

have improved the fake message efficacy *(FME)* [6, 7] which could be calculate as in below:

$$FME = \frac{\sum_{i=1}^{i=m} \frac{\overline{\omega}}{SI_i} - m}{\sum_{i=1}^{i=m} \frac{\overline{\omega}}{SI_i}} \qquad (6.15)$$

Where $\overline{\omega}$ is the big interval value, (SI_i) the subinterval value, (m) the total number of subintervals.

For example, if SN assigned a sequence $\omega_i \in$ [4,6,2,8], fake messages could be sent at the following subintervals [4,10,12,20,24,30,32 . . . . .], and the real messages could be sent at the following sub intervals [2, 4, 6, 8, 10, 12, ...]. By substituting $\omega_i = 8$, $SI_i = 2$ and m=4, FME will be 60%. Figure 6.7 exhibits the transmission of fake and real messages for two consecutive subintervals.

6.2.3. Energy conservation

When a node senses data or receives data that needs to be forwarded to the BS, it has to select the next one-hop node from the neighborhood set N_i. During the setup phase, each SN_i has information about the hop-distance for each of the neighbors stored in its table T_i. Typically, there are three sets: one set where the hop-distance is less than its own (uplink set), a set where the hop-distance equals to itself (equal-link set) and a set where the hop-distance is larger than itself (downlink set). Choosing a node randomly or by round-robin from the uplink set will be ideal in terms of delays since it will give the shortest path to the BS. However, that will cause the nodes in this set to consume more energy compared to the other two sets of the neighbors. After each transmission, the SN consumes some energy.

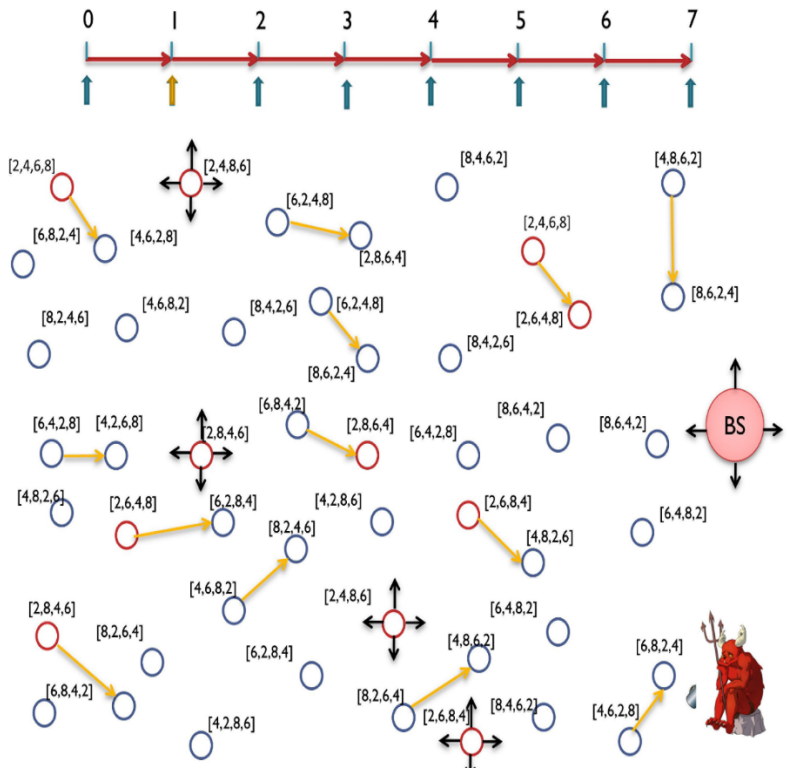


Figure 6.8: Any node assigned subinterval $\omega_i = 2$ will send a fake message if it does not have a real message to report.

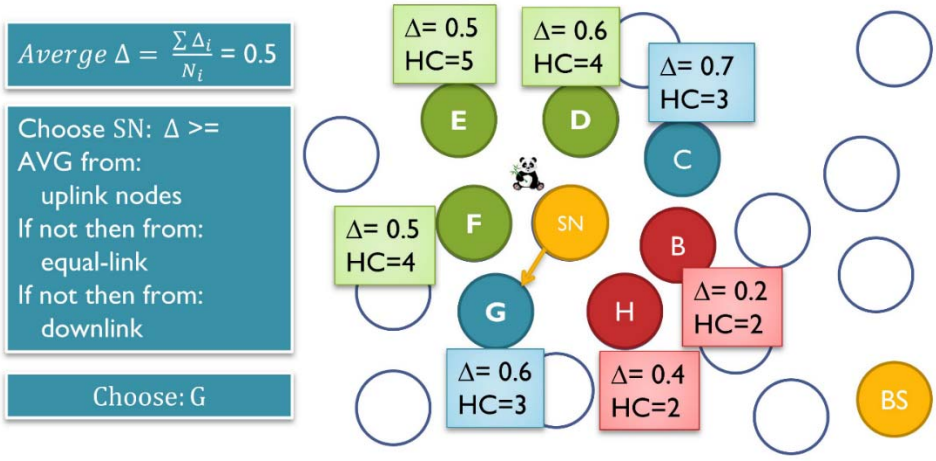


Figure 6.9: How to choose the forwarding node according to the energy levels of the neighbors.

The residual energy for SN_i will be calculated as below:

$$\Delta_{residual} = \frac{\Delta_{current}}{\Delta_{initial}} \qquad (6.16)$$

Each node will calculate its residual energy and share it with its one-hop neighbors. When the node sends fake messages, it will send its residual energy with it. The neighbor SN_j will store the value in its T_j for each of its neighboring nodes. This way, any sensor node will have some information about the residual energy level for its immediate neighbors. Figure 6.9 exhibits the mechanism for selecting the forward node.

6.2.4. Handling rate attack

One issue that WSN with one sink could suffer from is having higher transmission rate next to the BS where messages ultimately need to reach out to the BS as the final destination. In contrast, periphery sensors far from the BS could have light transmission rates. Figure 6.10 exhibits the issue. This could jeopardize the location privacy of the BS. One solution is to have multiple sinks distributed in the network. This contradicts with the pre assumptions we set for our framework so we will not address this solution in this work. The framework needs to maintain similar average rate among all the sensors. This could be achieved by increasing the number of fake messages transmitted by less busy nodes which means increasing the bandwidth usage and the power consumption. We need also to reduce the fake messages sent by busy nodes or delay the real messages to maintain similar rates. The latter is achieved automatically since the sensors do not send fake messages when they have real messages. However, this could be better tuned for average busy nodes as well. Having balanced rate in the WSN could help to maintain balanced average lifetime

for the nodes in the network. Presuming that all the nodes are heterogeneous in terms of energy would mean that busy nodes would be depleted sooner which could create an empty coverage area or a buffer zone between the sink and the peripheral SNs. This makes it a double fold problem. The first approach is to select a suitable location for the BS in the network map. Most of the literature shows a side location for the BS. It is maybe due to the fact that it is more suitable for the applications in hand where the BS is connected to the backbone network in a reachable area and sensors are unattended in out of reach areas. Figure 6.11 exhibits that the coverage area of a central BS is much better than a side BS. The density of nodes closer to the BS should be higher. The range of transmission for sensors in higher density areas may need adjustment to control energy dissipation. We could have multiple density areas around the BS where the density is reduced as it gets distant from the BS.

Figure 6.11 exhibits only two density areas for simplicity. If the storage of the sensor is not big enough which is unrealistic case with increasing storage technology in the sensors, the sensor does not need to include all the neighbors in the tables. The network will be divided into two areas, near (An) and far (Af). The framework will set average transmission rate (ATR) thresholds, Rmax and Rmin. Sensors in An will be loaded with Rmax where the sensors need to queue messages to maintain the threshold. In reverse, sensors in Af will be loaded with Rmin to maintain the lower threshold by sending more fake messages as needed. The sensor will calculate its average transmission rate over a period of time Tatr, which is preset by the framework.

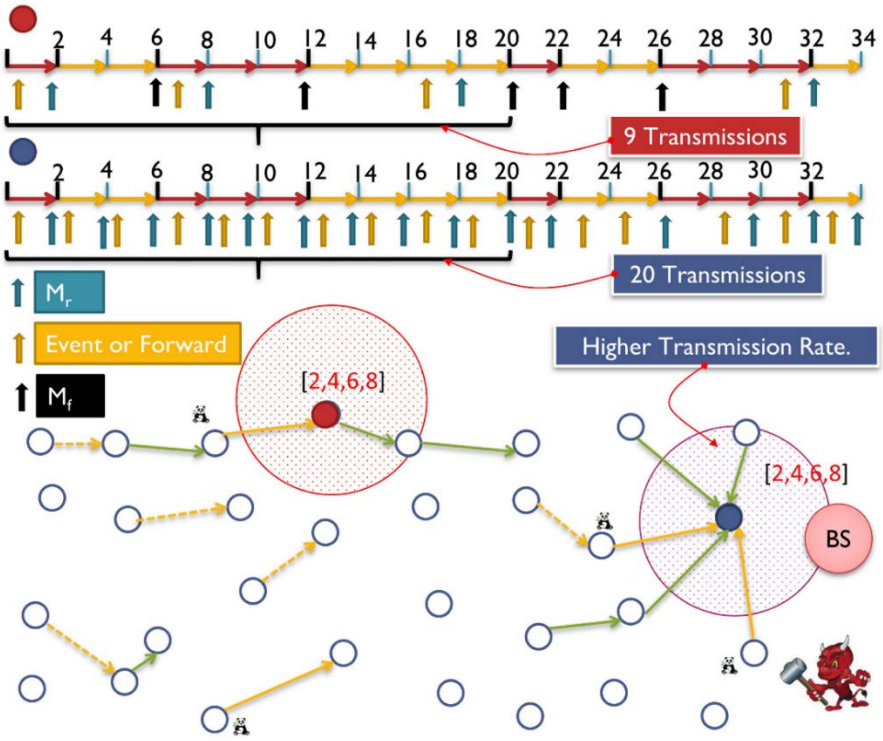


Figure 6.10: Higher transmission rate next to the BS [6].

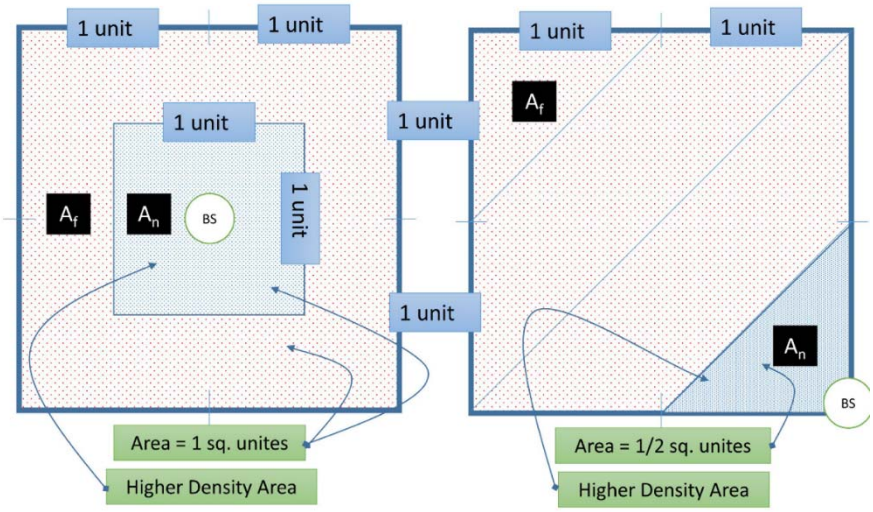


Figure 6.11: To balance the higher data rate nearby the sink, we acquire a higher density sensor distribution.

CHAPTER 7: ANONYMITY AND SECURITY ANALYSIS

We need to analyze FAC for both passive and active adversary attacks. The adversary (ADV) model has a global view of the network. ADV could target the source, intermediary and BS nodes. Usually, ADV starts by monitoring transmission somewhere in the network and then attempts to acquire sources (downlink direction) or BS (uplink direction). Passive attack is ordinarily the base for active attack. Once ADV determines the identity and location of a source or the BS, it consequently can launch various active attacks against certain nodes or disrupt the operation of the entire WSN. The main strength of passive ADV is the fact that neither SNs nor the BS will know about their existence. Nonetheless, active attacks can be detected if the framework instruments a reasonable IDS. Any comprehensive solution for location privacy should protect against anonymity attacks, temporal attacks and rate discovery attacks. We believe that routing privacy is useful only against local advisory and once the WSN faced with a global or a multi-local adversary, rounding privacy is not crucial. Thus, we have chosen short-path routing technique for this work. Any other routing protocols should be utilized to reduce delays and energy consumption.

7.1. Security against passive attacks

SNs use disposable pseudonyms to identify each other instead of using real IDs. No real ID stored in the sensor and no pseudonym is used more than once. Data is encrypted

all the way from the source to the BS using shared pair-wise keys. For *eavesdropping and content analysis,* ADV can intercept messages without being able to read them because data is encrypted all the way to the BS. The only information ADV can get from the captured messages is the pseudonyms: OHPID, BPID or FPID which are all temporary IDs and have no use except to calculate a new set of pseudonyms. Fortunately, the ADV cannot get from the captured messages, important parameters $a_{i\leftrightarrow j}$,b_i or c_i which are all required to calculated new pseudonyms. Source PIDs are all encrypted during transmission. For *hop-by-hop trace,* ADV can track a stream of messages from one node to another by overhearing the messages. The ADV will be challenged with many real and fake transmissions throughout the WSN. Furthermore, each node will retransmit the messages through different routes. For *size-correlation***,** ADV will be able to understand relationship between incoming and outgoing messages by analyzing sizes of the messages. This attack does not work for our framework since all the messages have commensurate size. For *identity correlation,* ADV cannot relate overheard identities to their nodes. It is not possible since SNs use different pseudonyms every time a message is transmitted. For *rate monitoring,* ADV tries to collect some statistical information about transmission rates. For instance, WSN will have a higher transmission rate nearby the sink. This is handled by issuing fake messages to maintain a similar transmission rate. For *angle-of-arrival (AoA),* ADV uses special hardware to determine the signal direction. The framework did not account for specific countermeasure, however, it becomes a more serious issue with mobile SNs. Furthermore, AoA would not perform well in our framework because of the uniform message distribution by using real and fake messages. For *received-signal-strength (RSS),* ADV uses special hardware to measure signal strength to calculate distance to the source.

This is not an issue for our framework since every transmitter has fixed transmission power and SNs are immobile.

7.2. Security against active attacks

In principle, we assume ADV knows encryption protocols used by the framework, however, the framework needs to hide *encryption keys* and *IDs*. Active attacks can be categorized into: *soft* and *hard*. For soft-active attacks, ADV tries to compromise SNs to get some information related to security of the sensors such as *keys* and *IDs*. Consequently, it will monitor all messages traversing through the compromised nodes to discover the source and the BS locations. ADV hides its presence by acting passively *(soft)* but once it captures privacy information, it reports the information to an external executer to do further damages (such as killing the Panda in the Panda game). For that, it is harder for the IDS to detect the attack. In hard-active attacks, ADV captures SNs and invasively forge messages, sent replay messages etc. Moreover, ADV could load powerful devices with the captured credentials to launch more catastrophic attacks. Hard-active attacks could be detected by IDS, however, it could depend very much on the sophistication of the IDS used. With that, it remains very challenging to countermeasure hard-active attacks. In the following two subsections, we will analyze the security of our framework against active attacks.

7.2.1. Soft-active attacks

If ADV *physically* compromises SN_i, then it captures two sets of information:

(i) Information related to the node itself: the current *PID*, the parameters used to calculate the pseudonyms, the hash functions, the keys and other information as listed in Table (2).

(ii) Information related to the neighbors as listed in Table (3).

The ADV would have all it needs to issue new valid pseudonyms and to send messages out to neighbors. Let's look closely at few scenarios.

Scenario 1: If ADV physically compromises SN_i, and if SN_j and $SN_r \in N_i$, so SN_i knows some information about both SN_j and SN_r. However, it cannot calculate important information such as $a_{j\leftrightarrow r}$ which is required for one-hop communication between SN_j and SN_r [68], because SN_i would need ID_j and ID_r which are both deleted permanently at the end of the setup phase. If SN_i hears a message, it cannot determine, with high confidence, the sender among neighbors while communicating with each other. If SN_i receives message from sources $\notin N_i$, then it would not be able to determine the source.

Scenario 2: If ADV *physically* compromises multiple SNs, let's call it set N_{cs}, and collects number of messages, let's call it set N_{cm}. Then, the number of compromised PIDs equal to N_{cm} since each message has unique PID. If the source $SN_i \notin N_{cs}$, then ADV cannot know the source node [7, 89].

Scenario 3: If the message sent by source SN_i as in *scenario 2* passes thought $SN_j \in N_{cs}$ or even through multiple compromised nodes, it will not be able to correlate the captured PID_i with SN_i.

Scenario 4: If a message sent by source SN_i and $\forall$ SN $\in N_i$ is also $\in N_{cs}$ (all neighbors are compromised), then ADV will be able to know that SN_i is the source. It is unrealistic situation to have many compromised nodes in one area. However, this proves that single or few compromised nodes cannot locate the identity of the source. In addition, a compromised node does not actually need to locate the sources within its range since it can detect the objects of interest (Panda) knowing that the ultimate goal of the adversary is to capture the *object* not the sensor reporting the object.

In summary, while we cannot prevent physical capturing of sensors, we need to make sure capturing sensors do not have destructive effects on other sensors. It is clear that our anonymity model protects against the *avalanche* or the *domino effect* behavior once one or few sensors are physically captured.

7.2.2. Hard-active attacks

If ADV physically compromises SN_i then it can launch denial of service attacks (DoS), which is an effort to temporarily or indefinitely suspend transmission in the network. It consumes the resources such as bandwidth, memory, storage, and processor time. When ADV compromises SNs, it would be able to send massive valid messages to consume system resources. The ADV will be able also to launch replay attacks where ADV gets credentials of the some sensors and attempts to mimic the sensors to send messages to other neighbors. The other attacks such as, forging attack, packet alternation, packet dropping and packet injection are all only possible to physically captured nodes. However, it cannot propagate easily behind neighbors. Nothing could be worse than having physically captured nodes where ADV has full control over the sensors. Good IDS can

detect such attacks and respond by removing the compromised nodes immediately. The most danger tactic of hard-active attacks is to prevent the real messages from following normal paths to the BS and force the messages to traverse through certain routes. Our main contribution to handle this attack is to put in place a seamless and efficient protocol to add and remove SNs while WSN in action.

7.3. Sink security

ADV can learn that a sensor has received a message in two ways: (*i)* When the sensor retransmits the message, which was tracked beforehand to another sensor, (*ii)* the ADV is able to make a correlation between the captured ID and the physical recipient sensor. The adversary cannot locate the BS location by compromising only one neighboring sensor because each transmission uses a different pseudonym. It actually will need to compromise multiple colluding sensors along the path to the BS or many neighbors of the BS. While we cannot prevent having many physically fallen sensors, our framework's goal is to delay the capturing of the BS provided that there are many colluding captured sensors in the WSN. A very interesting *scenario* is *exhibited* in Figure 7.1.

Let us presume $SN_r \in N_{cs}$. It issues a message with D_{bomb} such that: $APID_r$ || $OHPID_{r \leftrightarrow u}$ || $E_{r \rightarrow u}$ ($APID_r$ || PID_r || $E_{kr \leftrightarrow bs}$ $\boldsymbol{(D_{bomb})}$)**.** IF ADV compromise multiple nodes along the path to the BS where each sensor decrypts the data to read this *signature* at every hop: $\boldsymbol{(PID_r}$ **||** $\boldsymbol{E_{kr \leftrightarrow bs}}$ $\boldsymbol{(D_{bomb})}$**.** Providing the colluding sensors, in the path to the BS, read similar signature while it knows by design that every message should be directed uplink to the BS, the ADV could follow through to the BS. Having multiple compromised paths (with compromised sensors) reading the same pattern will give adversary more clues.

Compromised nodes can even collude to force the real messages to route through fixed suspected areas in effort to focus the capturing process which becomes a function of: (i) the size of the network, (ii) The traffic density, (iii) the number of compromised nodes. To solve this issue, we have to wipe out the signature before each transmission. Thus, every message will be forwarded to the next hop as below:

$$M_{u\rightarrow x} = APID_u \;||\; OHPID_{u\leftrightarrow x} \;||\; E_{u\rightarrow x}\,(APID_u \;||\; PID_r \;||\; \mathbf{PID_u} \;||\; \mathbf{E_{ku\leftrightarrow bs}}\,(E_{kr\leftrightarrow bs}\,(D_i)))$$

(7.1)

We have added a multiple levels of encryption which will be done at every hop using the shared key between the hop and the BS. In addition, *PID* of the hop will be added in sequence so the BS can do the decryption in sequence. This solution increases the *size* of the message proportionally to the number of hops. We suggest to have the onion encryption done for a distance of few hops, O_h. So, if $O_h = 2$, then we have only two extra encryptions. In addition, we need to account for O_h *PID's* added to the message.

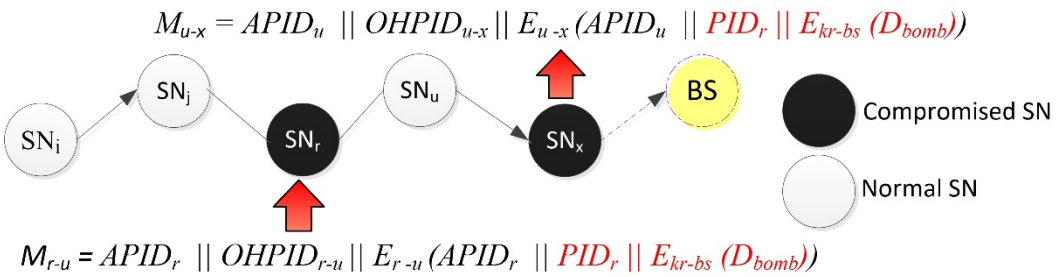


Figure 7.1: Hard-active Attack tries to get the BS by inserting a signature in the transmitted message.

We have implemented a WSN of 100 SNs uniformly distributed over 30 x 30 area where the average distance between the SNs 3.7 as in Figure 6.5. A random compromised sensors were interested in the network. We have simulated for O_h equals to 1, 2 and 3. The adversary succeeds when it knows all the nodes forming the curve around the BS in 1, 2 and 3 hops way, consecutively. Figure 7.2 exhibits the average number of transmissions required before the adversary can succeed. It is clear that with higher value for O_h, the network will be able to send more messages before the BS is compromised. Having a higher number of compromised nodes in the WSN will make it faster to capture the BS, as well.

7.4. Link anonymity

Link anonymity is to prevent the ADV from knowing the relationship between the sender and the receiver. If a message leaves a sender and leaves again the recipient as it is, the ADV would know the relationship between the two nodes. This is secured in our framework since every message is completely changed after each retransmission including the IDs. In addition, it maintains fixed size. Applying different delays and different next-hop direction should also increase the privacy of the link. Furthermore, the adversary cannot know if the link carries real or fake data.

7.5. Timing privacy

By using fake messages at variable interval times and message delays, it becomes super hard for the ADV to correlated messages being transmitted over the network as exhibited in Figure 6.5.

7.6. Routing Privacy

Although short-path routing is used in this framework, choosing the next hop is done according to certain *probabilistic* algorithm which accounts for the residual energy of the sensors and the usage frequency to increase the route privacy, as exhibited in Figure 6.9. ADV cannot relate routes to nodes due to the triple anonymity. Even if two messages for one sensor follow the exact same route, ADV will see them as if they are two different routes since each hop along the route carries messages with different PIDs.

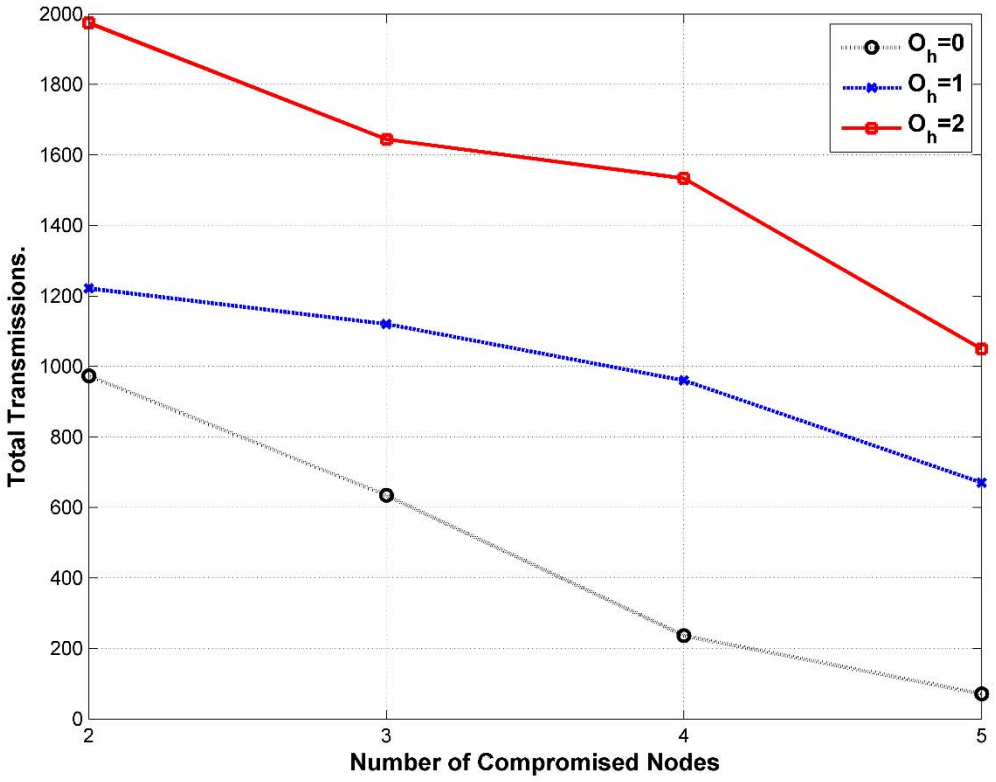


Figure 7.2: Protecting the BS by having onion encryption.

7.7. Data Privacy

All the data is encrypted before transmission and encrypted again at every hop along the route to the BS. Data will be authenticated by a message digest. The only time data is not protected when the sensors are physically compromised. The compromised nodes are able to inject data in the network. If ADV uses the compromised nodes actively, a good IDS can detect the falsified data. The framework provides a secure facility to remove compromised sensors and to add valid sensors, if needed, to the WSN.

7.8. SLP and BLP

SLP and BLP are achieved at first by having the triple anonymity *(source, BS, link)* which was argued earlier. ADV cannot infer any information from the intercepted messages. Passive attacks will not endanger the location privacy. However, strong active attacks could hinder the location privacy without having good IDS. Secondly, we have provided a solution for temporal privacy using ECAT. Thirdly, we have provided a solution for rate attacks. The three security measures will work hand in hand to provide location privacy.

7.9. Entropy

For the best case for anonymity, the adversary sees every sensor node sending a message at a fixed data rate at any one time. It also cannot distinguish any message from the rest of the messages in the network due to the implementation of ID anonymity solution. If we have N nodes in the WSN, the probability that one adversary can locate the sending node or the BS equals to:

$$P_i = \frac{1}{N} \qquad (7.2)$$

One of the methods to quantify for the degree of anonymity is calculating the rational entropy as expressed below:

$$\mathrm{EN(X)} = -\sum_{i=1}^{N}[P_i \times \log_2 P_i] \qquad (7.3)$$

The maximum entropy is when the adversary believes every sensor node has the same probability to be the transmitter (uniform distribution):

$$\mathrm{EN_{max}} = -\sum_{\mathrm{i=1}}^{\mathrm{N}}[\frac{1}{\mathrm{N}} \times \log_2 \frac{1}{\mathrm{N}}] = \log_2 \mathrm{N} \qquad (7.4)$$

Where $\sum_{i=1}^{N} P_i = 1$, P_i is the probability that the sensor node is the transmitting node. To calculate the degree of anonymity (A):

$$\mathrm{A} = 1 - \frac{\mathrm{EN_{max}} - \mathrm{EN(X)}}{\mathrm{EN_{max}}} = \frac{\mathrm{EN(X)}}{\mathrm{EN_{max}}} \qquad (7.5)$$

If all the nodes would send messages at every interval then the anonymity:

$$\mathrm{A} = \frac{\mathrm{EN(X)}}{\mathrm{EN_{max}}} = \frac{\log_2 \mathrm{N}}{\log_2 \mathrm{N}} = 1 \qquad (7.6)$$

However, it is not realistic to have transmissions by all N in one time. If we presume the minimum number of transmissions per one interval (N_{min}) and the average transmissions per one interval (N_r), then anonymity ranges:

$$\mathrm{A} = \frac{\log_2 \mathrm{N_{min}}}{\log_2 \mathrm{N}} < \frac{\log_2 \mathrm{N_r}}{\log_2 \mathrm{N}} < 1 \qquad (7.7)$$

With the presence of global adversary the anonymity will range $0 > A \geq 1$ depending on the number of node transmitting (N_t) at one interval time as exhibited in

Figure 7.3. However, local adversary can detect transmission within its range, let's say 50 nodes density. If the transmissions within the adversary range is more than 50 transmissions, then it would not contribute to the anonymity against that particular adversary. The fake messages transmission could be reduced to have all the transmissions (real and fake) equals to 50.

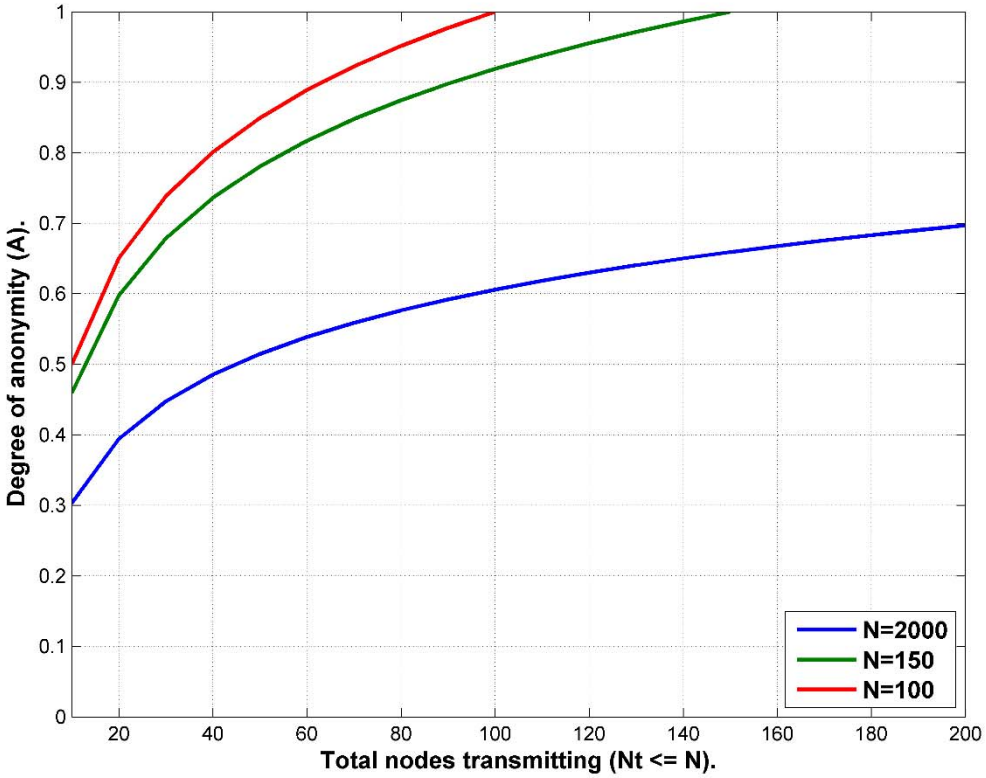


Figure 7.3: Degree of anonymity (A).

CHAPTER 8: PERFORMANCE EVALUATION

In this section, we evaluate the performance of the FAC framework, including delays, energy dissipation, data rate privacy, storage, processing, computational, and communication costs.

8.1. Delay:

In SGAT, sensors transmit/forward the data at the end of the interval, which would cause a huge delay considering the volume of messages that each sensor needs to transmit during the network real-time operation. In addition, the messages traverse through multiple hops until it gets to the BS, which makes the accumulated delays significant. The other alternative scheme is having the sensors select one of the following subintervals (ω) randomly to forward the message. This also will cause some unnecessary delays although it could help in hiding the temporal behavior of the sensors. ECAT scheme divides ($\overline{\omega}$) into subintervals (ω), so the transmission will happen at the first available subinterval when the message is ready. We have simulated a smaller network to the one descripted in Section 8.3 for the transmission delays. It includes 48 SNs only with ω distribution as presented in Figure 6.5. We have three simulations using SGAT, ECAT, and random delays. Figure 8.1 shows that delay per one-transmission increases throughout the network as the number of transmitted messages increases which could cause unjustifiable delays especially in the real time applications. Figure 8.2 also shows the average delays for the three schemes. It shows that using ECAT has improved delays by 64% compared to SGAT. The total delay

for one message from a source to a destination (BS) is calculated according to expression Equation (37). It is a function of the distance from the BS (hc) which we technically have no control over after sensors deployment. In addition, it is a function of the chosen ($\overline{\omega}$) and (ω) values for the system. The larger the (ω), the more delays accumulated. We have simulated the same network using ECAT for the total delay as exhibited in Figure 8.3. It shows that the delay rises as the *hc* increases and as the size of the intervals widens. We conclude of these simulations that the performance of ECAT is better than SGAT while it continues to provide a good temporal privacy. Using a fixed delay will reduces the delays slightly but it provides a very week temporal privacy.

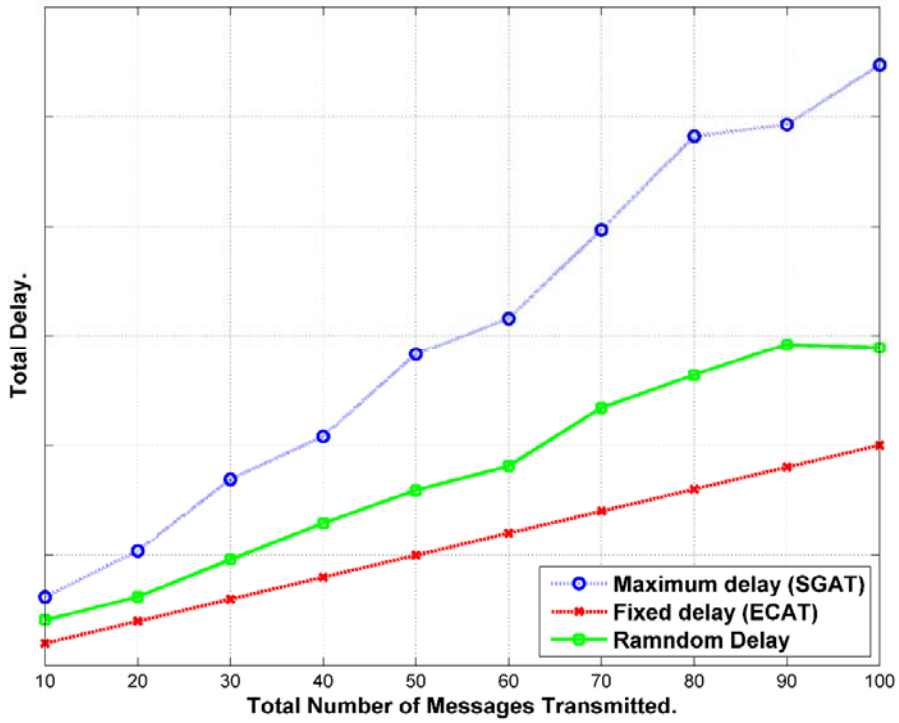


Figure 8.1 : Total accumulated delay per one node.

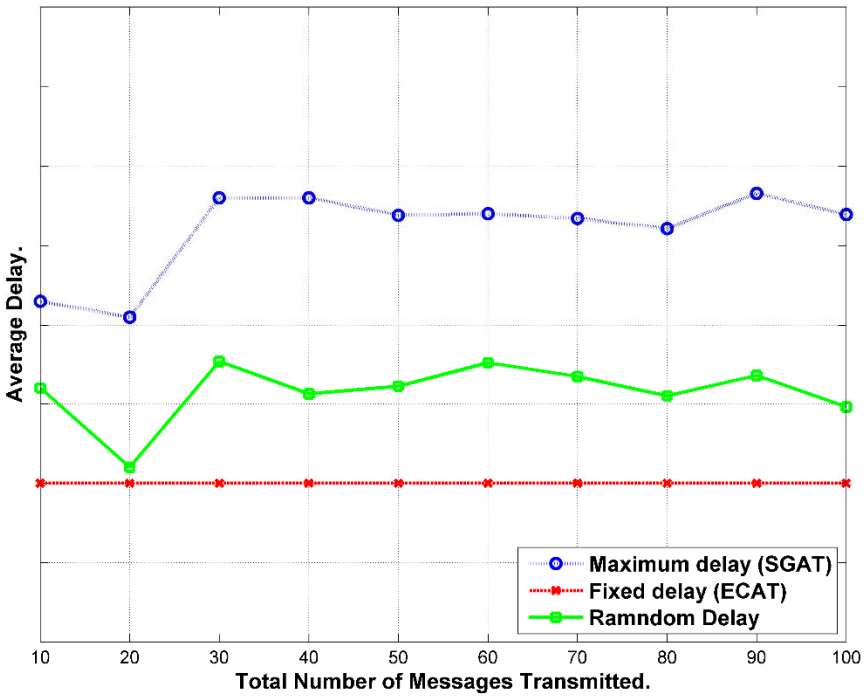


Figure 8.2: Average accumulated delay per one node.

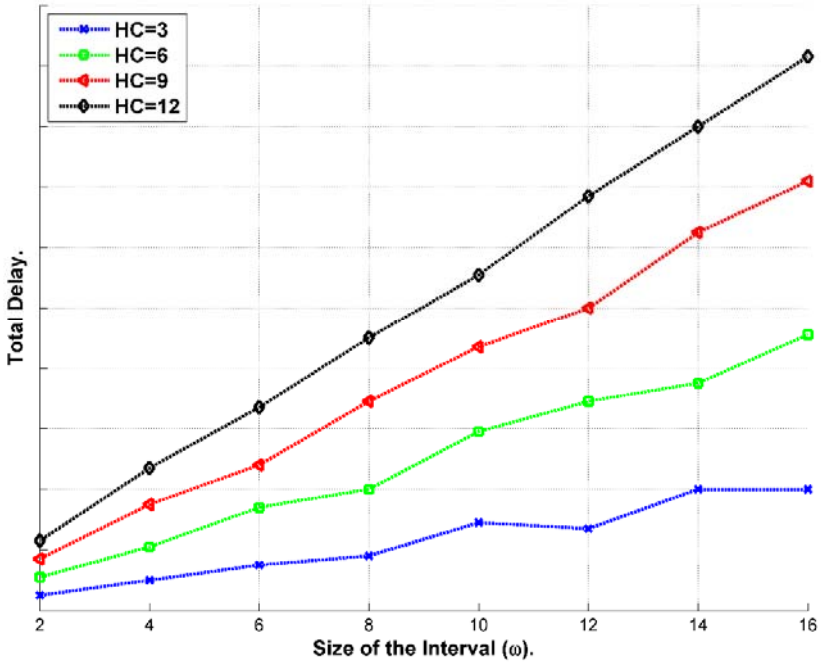


Figure 8.3: The accumulated delay is a function of the hop count (hc) and the size of ω.

8.2. Energy cost

In our work, we will assume a simple energy dissipation model [5-7, 90]. The radio dissipates Ɛ nJ/bit for both transmission and reception by the sensors circuitry. In addition, it consumes ε nJ/bit/m^2 for the transmitter amplifier to achieve an acceptable signal to noise ratio. Therefore, to transmit *k* bits for *r* distance, the total transmission energy dissipation will be:

$$E_t = k * Ɛ + k * r^2 * ε \qquad (8.1)$$

In addition, the receiver would consume for reception of k-bit message:

$$E_r = k * Ɛ \qquad (8.2)$$

SGAT assumes that every node would send one message at the end of each interval where the message could be either real or fake. If we have *N* nodes in the network, then we expect *N* messages during each interval. The energy spent for transmission or reception is similar per one message because we have unified-size messages to prevent size correlation attacks by the adversary. If we have *p* percent of the nodes issue or forward real data at each interval, then *1-p* percent of the energy and the bandwidth is wasted on fake messages. We can adjust the amount of energy consumed by increasing the interval period *(ω)*. However, increasing *(ω)*, would increase the delay. The consumption of transmitting fake messages is a double fold since the transmitter will consume E_t for every message and all the neighbors (N_i) will consume($N_i * E_r$). When the transmission range increases, N_i increases. The total energy consumed in the network to send real and fake messages in one interval [90]:

$$E_R = (N)(k * \mathcal{E} + k * r^2 * \varepsilon) + (N * N_i)(k * \mathcal{E}) \quad (8.3)$$

ECAT has improved the energy dissipation by reducing the amount of fake messages transmitted while maintaining the required temporal security. The number of total messages transmitted per interval has reduced from 100% to a certain percentage *(p)*. We have simulated the WSN in Figure 6.8 presented in section 8.2 to calculate the energy dissipation. Figure 8.4 exhibits the total energy dissipation per one message considering the transmitter, the recipients and the range of transmission. The size of the messages is *8,000 bits,* $\mathcal{E}$ is *50 nJ/bit* and ε is *100 pJ/bit/m*^{2}. The simulation shows that the energy dissipation due to the increase of sensor range is marginal compared to the increase in energy dissipation due to the increase of neighbors *(N_i)*. However, increasing the range could increase N_i if the WSN has uniform sensor distribution. The technology has changed over years and the consumption by the sensor circuit for sending and receiving is improving [91]. We have also simulated the network to see how the transmission of fake messages has improved using ECAT. Figure 8.5 exhibits the simulation of *40* subintervals ***(ω)***. The graph shows the maximum possible fake message at each subinterval ***(ω)***. For instance, the total fake messages during *ω=10* is *16* messages while during *ω*=32 is *20* messages. The mean of fake transmissions is 19.5 (compared to *48* messages in SGAT). The average fake messages for the whole simulated period is 19.5 messages which shows about 59% reduction of possible fake messages comparted to SGAT. The number of fake message will be reduced further as the network gets busy transmitting real messages since a sensor node don't send a fake message at a subinterval where it has a real message to convey. We have simulated the same network with 70% probability of event occurrence. Figure 8.6 shows

that the average fake messages has reduced to 5.85 messages, which is almost 88% reduction from SGAT. This also will reduce the energy consumption significantly.

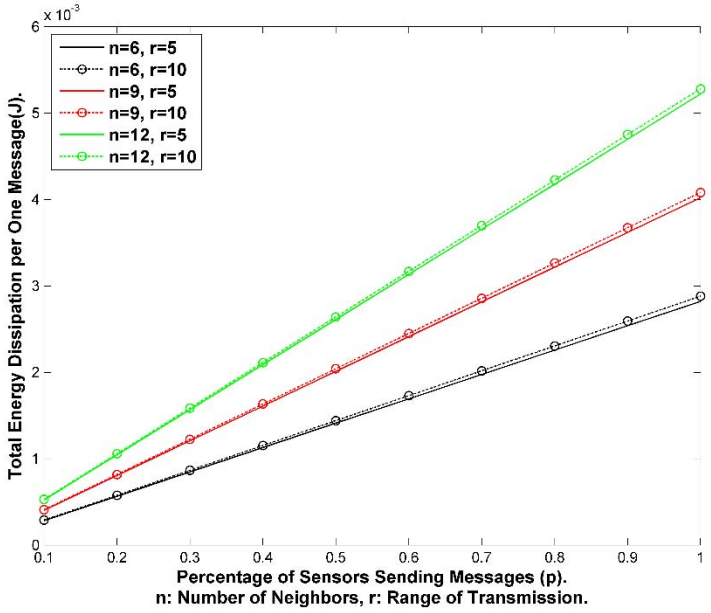


Figure 8.4: The energy dissipation increases as the number of neighbors and the sensor transmission range increases.

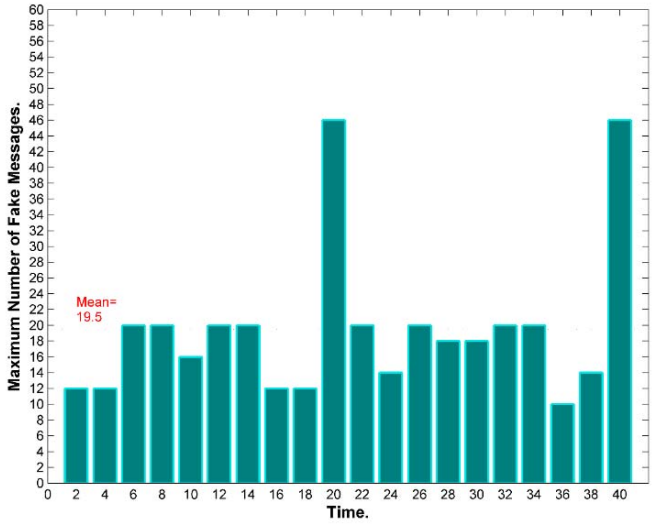


Figure 8.5: A simulation for the maximum possible fake messages per subinterval using ECAT.

The most expensive operation for energy consumption is transmission of *bits* from one node to another. We use two stages for air communication in our framework, (i) In the setup phase and, (ii) in the communication phase. The data transmission during setup phase is minimal. During communication phase, data will be forwarded hop-by-hop to the BS. Every packet is equally sized to prevent time and size correlation. We have introduced a probabilistic fake message transmission scheme which none of the other protocols adopted. Real messages are sent at the end of each subinterval time to prevent delays.

The cost per message at one interval time is:

$$\text{Average Message Cost} = \frac{R+(N-R)P_r+A}{R} \qquad (8.4)$$

Where R is the total number of SNs sending real messages at one subinterval time, P_r the probability of sending fake message by SNs, and A is the average number of acknowledgements in one interval. None of the other schemes addressed the issue of rate analysis attacks which is one of the easiest attacks any adversary can use. Using fake messages is an expensive solution. However, we have designed FAC to be adaptive to the network traffic situation by using a closed-loop system. The sink can always increase or decrease the amount of fake messages used according to the reports it is getting about the system security. The threshold values of R_{min} and R_{max} are also adjustable according to the network situation.

8.3. Transmission Rate Privacy

To handle this issue, we have adopted two threshold values: R_{min} and R_{max} where the sensor needs to keep its message transmission rate between these two values. The

sensor needs to send real message at the end of subinterval time slot. If it does not have a real message, then it needs to send a fake message only if that time slot is scheduled to send a fake message according to ECAT protocol, otherwise no transmission will happen and the slot remains idle. Ideally, the sensor has a real message at the subinterval so it does not need to waste a slot by sending a fake message.

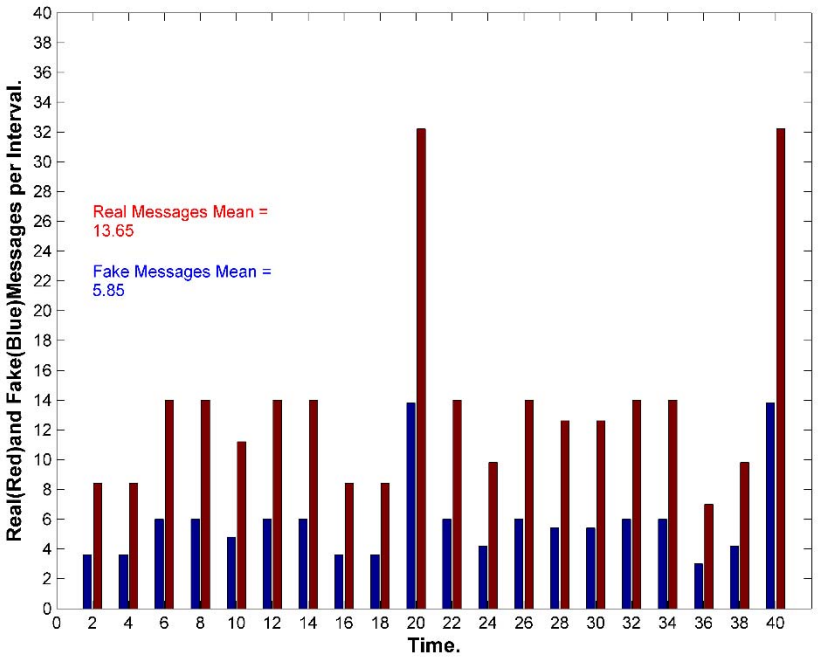


Figure 8.6: The average fake messages in a busy network with 70% of the slots occupied by real messages.

The sensor can use this facility to control the threshold data rate. For instance, if the rate is high (such as in areas nearby the BS), it can replace fake messages with real messages which is a double fold beneficial. If all the fake messages are already replaced and still there is real messages above the threshold, then the sensor is required to queue the messages and delay the transmission to maintain same average message rate between R_{min}

and $R_{max.}$ In contrast, if the message rate is low (as in the periphery sensors), then the sensor can transmit more fake messages during idle slots. We have simulated the network in Figure 6.5 and assigned the R_{min} to be *thirteen* messages for two consecutive ($\overline{\omega}$) intervals (a total of 20 subintervals). We have assigned the R_{max} to be 13 messages during this period, which is 7 less than the total number of subintervals. That means we allowe up to 13 real and fake messages during these two consecutive ($\overline{\omega}$) intervals. Figure 8.7 exhibits the output of the simulation for four different individual transmissions. For instance, the first transmission shows, *14* real messages (blue bar), *2* fake messages (light blue bar), and *4* idle slots (green bar). The total real and fake messages is *16 (orange bar)* which is above the assigned threshold, *thirteen*, by *three* messages, which is expressed by the brown bar.

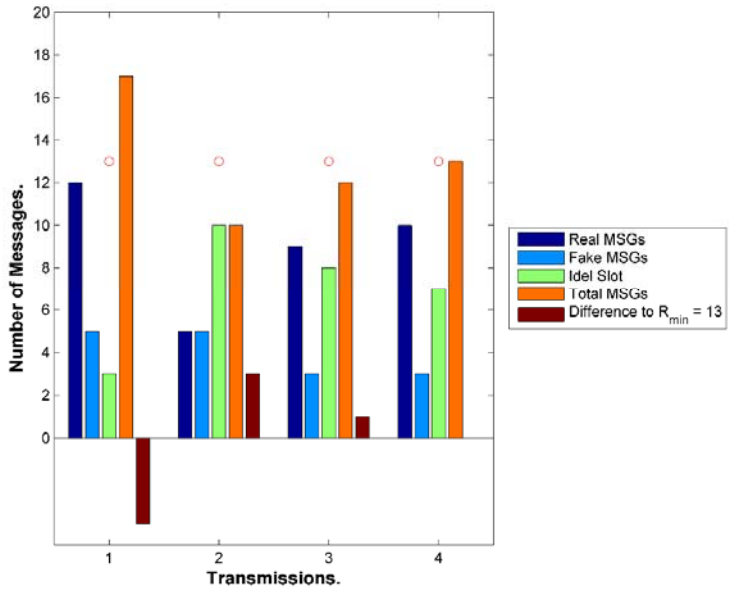


Figure 8.7: The simulation shows the total real messages, fake messages and idle slots.

The SN will cancel *three* fake messages out of the *four*. In some worse cases, the sensors would need to queue the messages for next slots. For instance, if there is 15 real messages during this time, the system send only 13 messages and queue 2 messages for the next period. Ultimately, the overall message rate during T_{atr} will be within the assigned thresholds.

We have simulated this approach extensively for *(500), (1,000), (1,500)* and *(2,000)* transmissions as exhibited in Figure 8.8. The first, third and fifth bar sets show the total amount of fake messages needed to be replaced with real messages to maintain R_{min} for the thresholds of th=10, th=11 and th=12 consecutively. For instance, a threshold of 10 means that the maximum number of messages transmitted should be 10 (out of 20 in our simulation). The second, fourth and sixth bar sets also show the number of messages which needed to be queued for the three consecutive threshold values. So, if we have real messages above the number of scheduled fake messages, then we have to queue the messages for the next period of T_{atr} . Overall, this simulation exhibits a great preference since we always would like to reduce the amount of fake messages and keep the bandwidth busy with real messages whenever it is possible. In addition, the simulation exhibits very small messages need to be queued (delayed). It shows as we increase the threshold value the less fake messages replacement or delays is required. In summation, reducing the fake messages and keeping the delayed message minimal is the goal which ECAT clearly achieves.

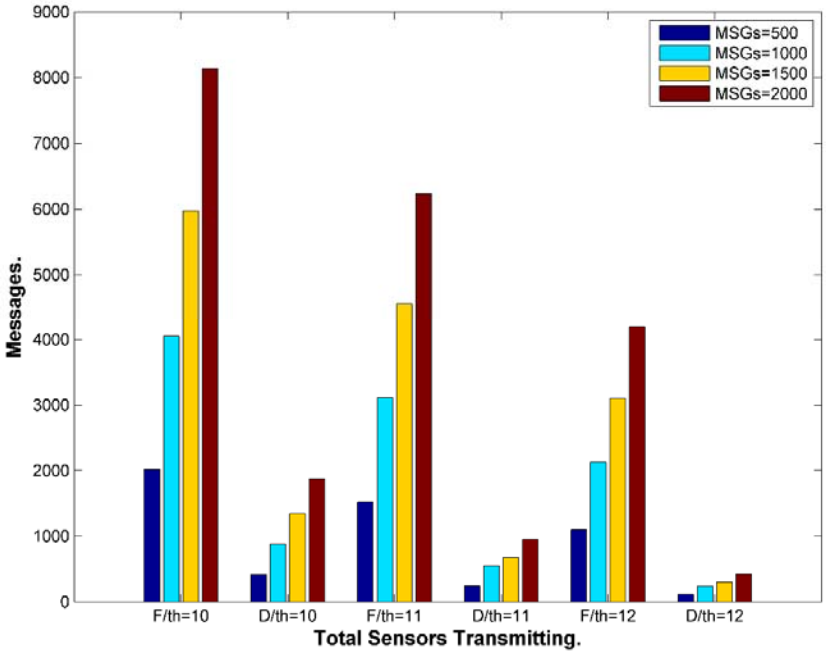


Figure 8.8: Simulation for busy network with different Rmin threshold values th=10, 11 and 12.

8.4. Storage Evaluation

There are two sets of information stored in a SN_i: (i) information related to the sensor itself such as random numbers: (a_i, b_i, c_i), pseudonyms: (PID_i, $BPID_i$, $FPID_i$, $APID_i$), keys: ($ki \leftrightarrow bs$, bki, fbki), (ii) information related to neighbors which include, random numbers: ($a_{i\leftrightarrow j}$, b_j, c_j), pseudonyms: ($OHPID_{i\leftrightarrow j}$, $BPID_j$, $FPID_j$), keys: ($k_{i\leftrightarrow j}$), Misc: ($link_{i\leftrightarrow j}$, Δ_j).

If we presume that the keys, the random numbers, the pseudonyms and the hash functions are all *n* bits long in average, and the required bits for miscellaneous data altogether is *two* bytes, and the average number of neighbors N_{ave}, then the total storage memory required is:

$$\text{Storage} = 10n + (7n+16)*N_{ave} \qquad (8.5)$$

Chen et al. [68] indicated the storage for *SAS* , CAS, *APR, DCARPS and EAC.* We also calculated the storage for *PhID, ACS*, HIR and RHIR. All are listed in Table 4. The size of storage increases proportionally when the size of *n* increases. The most common hashing functions which are considered very secure are: *MD4*[92] which uses 128-bits digest, *SHA-1*[92] which uses 160-bits digest, and *Whirlpool* [92] which uses 512-bits digest [92].

Table 8.1: Performance comparison.

No	**Scheme**	**Storage cost (bits)**	**Computation cost**
1	SAS	$2nN + 4nN_{ave} + 16$	No hashing operations
2	CAS	$6n + 7nN_{ave} + 16$	Two hashing orations and two encryptions
3	HIR	$2n+2nN_{ave}$	One hashing function
4	RHIR	$2n+2nN_{ave}+nkN_{ave}$	No hashing functions
5	APR	$9n + 7nN_{ave} + 2N - 2N_{ave} - 2$	Six hashing functions
6	DCARPS	3n	No hashing functions
7	ACS	$5nN_{ave}$	Two hashing functions
8	PhID	$(3n+2)* N_{ave}$	Four hashing function
9	EAC	$6n + 6nN_{ave} + 2$	Four hashing operations
10	FAC	$10n + (7n+16)*N_{ave}$	Four hashing operations & O_h encryptions

8.5. Processing and Computational Evaluation

Hash functions are used to calculate the pseudonyms and symmetric cryptography is used to encrypt the messages. Because we need to calculate three pseudonyms and one acknowledgement pseudonym after each transmission, using encryption to create pseudonyms was avoided since it requires more processing power compared to hash functions. When a SN senses data, it needs to calculate four OWH for PID, OHPID, and APID at the sender and OHPID at the receiver. If the system opts for data authentication, then another hash function is needed. The source node needs only one encryption for the data if O_h=0, however, it needs O_h more encryptions if onion fashion is used. Each intermediary node needs one decryption operation and then another encryption to issue the new message. Chen et al. [68] indicates that SAS does not use hashing or encryption to create pseudonyms because it uses already created pseudonyms from a space. The other scheme by Chen et al. [68], CAS, uses two hashing operations and two encryption operations. APR uses at least six hashing functions. DCARPS uses constant IDs, so no hashing functions or encryptions for creating IDs. EAC has four hashing operations. It is clear that our framework needs a bit extra processing power due to the higher privacy and security we have achieved. None of the other schemes can achieve privacy against global threats and active adversary attacks. The power consumption due to the additional encryption operations is marginal compared to the power consumption caused by data transmission. Figure 8.9 exhibits different storage size for different privacy schemes which are discussed throughout this work. It shows that the increase in storage is *linear* and relatively comparable to the other protocols. The size of the storage would increase when

the number of neighbors increases. Each SN has limited flash memory size which could confine the maximum number of neighbors that a sensor can fit. As an example, *TelosB* mote [54, 68] has *1MB* external flash memory. Thus, if one neighbor node requires 1.2k bits of storage memory, then TelosB could fit more than 800 neighbors which is way much more than what is needed in practical networks. Although FAC shows a bit of increase in the storage required to store the pseudonyms but it is the only one, among the discussed protocols in this work, provides a steady and functional anonymity and location privacy under strong global and active attack. In addition, the current technology provides sensors with sizable storage memory which makes it not an issue at all.

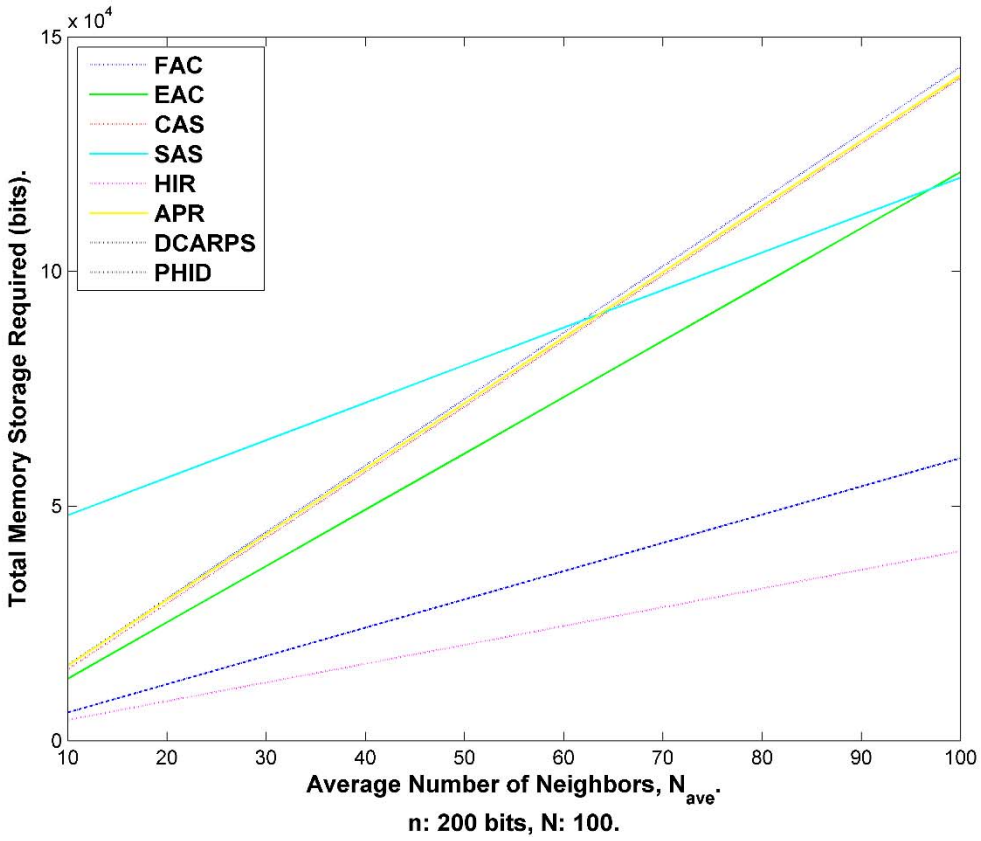


Figure 8.9: Size of storage memory using different privacy schemes.

CHAPTER 9: CONCLUSIONS AND FUTURE WORK

FAC is a modular framework that provides source, link and sink anonymity. It also provides temporal privacy and rate privacy. None of the previous related-work have a comprehensive solution for anonymity and location privacy. The three modules provided in FAC are made work together to prevent any statistical analysis attacks. The quadruple privacy (anonymity, temporal, rate, statistical) has provided a fortified SLP and BLP. FAC has addressed both local and global adversary. We have used a complex anonymity module where pseudonyms to replace real IDs are used. To provide temporal privacy both delays and fake messages are used. The use of fake messages was adjusted to manage the energy consumption. Two schemes are introduced, SGAT and ECAT. FAC is able to handle both homogenous and heterogeneous sensor nodes. FAC is both energy-aware and delay-aware. We have demonstrated that FAC can withstand passive and active attacks by presenting scenarios and provided solutions. The memory cost was mathematically analyzed for the framework. The computational complexity for encryptions and hash functions was analyzed. To provide security for the BS against colluding active attacks, we have introduced onion encryptions. We have simulated the performance of the framework. The future work would include enhancement on the fake messages probabilistic scheme. In addition, we will implement FAC for different routing protocols such as clustered networks. We would plug FAC in some civil and military applications for further analysis, development and improvement. One of the issues that any researcher in the area could face

is the lack of benchmarks to test the new proposed algorithms and protocols. There is a lack of standards and definitions for many terms. As an example, there is to specific definition for the global adversary. We do not know how feasible is having such an adversary in very large networks. There is no specific definition for how it works and what kind of timing it needs to capture a sensor node. The definitions of global, local, multi-local adversary view are vague. There is no standard routing algorithms. The security and privacy of clustered networks is not often discussed. We believe that the scholars of this area need to cooperate and come up with some standard checks and test beds that future work needs to consider.

REFERENCES

[1] Z. A. Eu, H.-P. Tan, and W. K. G. Seah, "Design and performance analysis of MAC schemes for Wireless Sensor Networks Powered by Ambient Energy Harvesting," *Ad Hoc Networks,2011,* vol. 9, pp. 300-323.

[2] D. K. Noh and J. Hur, "Using a dynamic backbone for efficient data delivery in solar-powered WSNs," *Journal of Network and Computer Applications, 2012* vol. 35, pp. 1277-1284.

[3] Y. Li and J. Ren, "Providing Source-Location Privacy in Wireless Sensor Networks," in *Wireless Algorithms, Systems, and Applications*. vol. 5682, B. Liu, A. Bestavros, D.-Z. Du, and J. Wang, Eds., ed: Springer Berlin Heidelberg, 2009, pp. 338-347.

[4] M. Conti, J. Willemsen, and B. Crispo, "Providing Source Location Privacy in Wireless Sensor Networks: A Survey," *Communications Surveys & Tutorials, IEEE,* 2013, vol. 15, pp. 1238-1280.

[5] M. A. Abuhelaleh, T. M. Mismar, and A. A. Abuzneid, "Armor-LEACH - Energy Efficient, Secure Wireless Networks Communication," in *Proceedings of 17th International Conference on Computer Communications and Networks. ICCCN.*, 2008, pp. 1-7.

[6] A. Abuzneid, T. Sobh, and M. Faezipour, "Temporal Privacy Scheme for End-to-End Location Privacy in Wireless Sensor Networks " presented at the Electrical, Electronics, Signals, Communiction & optimization EESCO-2015, pp 1-6.

[7] A. Abuzneid, T. Sobh, and M. Faezipour, "An Enhanced Communication Protocol for Anonymity and Location Privacy in WSN," presented at the IEEE WCNC 2015, 2015 pp, 1-6.

[8] A.-S. Abuzneid, T. Sobh, M. Faezipour, A. Mahmood, and J. James, "Fortified Anonymous Communication Protocol for Location Privacy in WSN: A Modular Approach," *Sensors,* 2015, vol. 15, pp. 5820-5864.

[9] A. A. Cardenas, T. Roosta, and S. Sastry, "Rethinking security properties, threat models, and the design space in sensor networks: A case study in SCADA systems," *Ad Hoc Networks,* 2009, vol. 7, pp. 1434-1447.

[10] Y. Ouyang, Z. Le, D. Liu, J. Ford, and F. Makedon, "Source location privacy against laptop-class attacks in sensor networks," presented at the Proceedings of the 4th international conference on Security and privacy in communication netowrks, Istanbul, Turkey, 2008.

[11] A. Abbasi, A. Khonsari, and M. S. Talebi, "Source Location Anonymity for Sensor Networks," in *6th IEEE Consumer Communications and Networking Conference, CCNC* 2009, pp. 1-5.

[12] A. A. Nezhad, A. Miri, and D. Makrakis, "Location privacy and anonymity preserving routing for wireless sensor networks," *Computer Networks,* 2008, vol. 52, pp. 3433-3452.

[13] L. Yao, L. Kang, P. Shang, and G. Wu, "Protecting the sink location privacy in wireless sensor networks," *Personal and Ubiquitous Computing,* 2013, vol. 17, pp. 883-893, 2013/06/01.

[14] N. Li, N. Zhang, S. K. Das, and B. Thuraisingham, "Privacy preservation in wireless sensor networks: A state-of-the-art survey," *Ad Hoc Networks,* 2009, vol. 7, pp. 1501-1514.

[15] A. Pfitzmann and M. Köhntopp, "Anonymity, Unobservability, and Pseudonymity — A Proposal for Terminology," in *Designing Privacy Enhancing Technologies*. vol. 2009, H. Federrath, Ed., ed: Springer Berlin Heidelberg, 2001, pp. 1-9.

[16] P. Kamat, Y. Zhang, W. Trappe, and C. Ozturk, "Enhancing Source-Location Privacy in Sensor Network Routing," in *Distributed Computing Systems. ICDCS . Proceedings. 25th IEEE International Conference on*, 2005, pp. 599-608.

[17] C. Ozturk, Y. Zhang, W. Trappe, and M. Ott, "Source-location privacy for networks of energy-constrained sensors," in *Software Technologies for Future Embedded and Ubiquitous Systems. Proceedings. Second IEEE Workshop on*, 2004, pp. 68-72.

[18] D. Jing, R. Han, and S. Mishra, "Countermeasures Against Traffic Analysis Attacks in Wireless Sensor Networks," in *First International Conference on Security and Privacy for Emerging Areas in Communications Networks. SecureComm.* , 2005, pp. 113-126.

[19] J. Ying, C. Shigang, Z. Zhan, and Z. Liang, "A novel scheme for protecting receiver's location privacy in wireless sensor networks," *IEEE Transactions on Wireless Communications,* 2008, vol. 7, pp. 3769-3779.

[20] L. Xinfeng, W. Xiaoyuan, Z. Nan, W. Zhiguo, and G. Ming, "Enhanced Location Privacy Protection of Base Station in Wireless Sensor Networks," in *MSN. 5th*

International Conference on Mobile Ad-hoc and Sensor Networks., 2009, pp. 457-464.

[21] M. Raj, N. Li, D. Liu, M. Wright, and S. K. Das, "Using data mules to preserve source location privacy in Wireless Sensor Networks," *Pervasive and Mobile Computing,* 2014, vol. 11, pp. 244-260.

[22] R. A. J. Shaikh, Hassan; D'Auriol, Brian J.; Lee, Heejo; Lee, Sungyoung; Song, Young-Jae., "Achieving Network Level Privacy in Wireless Sensor Networks.," *Sensors,* 2010, vol. 10, pp. 1447-1472.

[23] S. Armenia, G. Morabito, and S. Palazzo, "Analysis of Location Privacy/Energy Efficiency Tradeoffs in Wireless Sensor Networks," in *NETWORKING 2007. Ad Hoc and Sensor Networks, Wireless Networks, Next Generation Internet*. vol. 4479, I. Akyildiz, R. Sivakumar, E. Ekici, J. Oliveira, and J. McNair, Eds., ed: Springer Berlin Heidelberg, 2007, pp. 215-226.

[24] Y. Jianbo and W. Guangjun, "Preserving Source-Location Privacy in Energy-Constrained Wireless Sensor Networks," in *Distributed Computing Systems Workshops, 2008. ICDCS '08. 28th International Conference on*, 2008, pp. 412-416.

[25] K. Lei, "Protecting Location Privacy in Large-Scale Wireless Sensor Networks," in *Communications, 2009. ICC '09. IEEE International Conference on*, 2009, pp. 1-6.

[26] P. Spachos, S. Liang, and D. Hatzinakos, "Opportunistic routing for enhanced source-location privacy in wireless sensor networks," in *Communications (QBSC), 2010 25th Biennial Symposium on*, 2010, pp. 315-318.

[27] W.-P. Wang, L. Chen, and J.-x. Wang, "A Source-Location Privacy Protocol in WSN Based on Locational Angle," in *Communications, 2008. ICC '08. IEEE International Conference on*, 2008, pp. 1630-1634.

[28] L. Xi, J. Xu, and P. Myong-Soon, "Location Privacy against Traffic Analysis Attacks in Wireless Sensor Networks," in *International Conference on Information Science and Applications (ICISA).* 2010, pp. 1-6.

[29] X. Yong, L. Schwiebert, and S. Weisong, "Preserving source location privacy in monitoring-based wireless sensor networks," in *Parallel and Distributed Processing Symposium,* 2006, p. 1-8.

[30] L. Yun and R. Jian, "Mixing Ring-Based Source-Location Privacy in Wireless Sensor Networks," in *Computer Communications and Networks, 2009. ICCCN 2009. Proceedings of 18th Internatonal Conference on*, 2009, pp. 1-6.

[31] L. Yun, L. Lightfoot, and R. Jian, "Routing-based source-location privacy protection in wireless sensor networks," in *Electro/Information Technology, 2009. eit '09. IEEE International Conference on*, 2009, pp. 29-34.

[32] L. Zhang, "A self-adjusting directed random walk approach for enhancing source-location privacy in sensor network routing," presented at the Proceedings of the 2006 international conference on Wireless communications and mobile computing, Vancouver, British Columbia, Canada, 2006.

[33] L. Lightfoot, L. Yun, and R. Jian, "Preserving Source-Location Privacy in Wireless Sensor Network Using STaR Routing," in *Global Telecommunications Conference (GLOBECOM 2010), 2010 IEEE*, 2010, pp. 1-5.

[34] H. Xiaoyan, W. Pu, K. Jiejun, Z. Qunwei, and L. jun, "Effective probabilistic approach protecting sensor traffic," in *IEEE Military Communications Conference. MILCOM.*, 2005, Vol. 1, pp. 169-175.

[35] P. Kamat, W. Xu, W. Trappe, and Y. Zhang, "Temporal privacy in wireless sensor networks: Theory and practice," *ACM Trans. Sen. Netw.*, 2009, vol. 5, pp. 1-24.

[36] K. Mehta, L. Donggang, and M. Wright, "Location Privacy in Sensor Networks Against a Global Eavesdropper," in *IEEE International Conference on Network Protocols. ICNP.*, 2007, pp. 314-323.

[37] S. Min, Y. Yi, Z. Sencun, and C. Guohong, "Towards Statistically Strong Source Anonymity for Sensor Networks," in *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE*, 2008.

[38] Y. Yang, M. Shao, S. Zhu, B. Urgaonkar, and G. Cao, "Towards event source unobservability with minimum network traffic in sensor networks," presented at the Proceedings of the first ACM conference on Wireless network security, Alexandria, VA, USA, 2008.

[39] R. Doomun, T. Hayajneh, P. Krishnamurthy, and D. Tipper, "SECLOUD: Source and Destination Seclusion Using Clouds for wireless ad hoc networks," in *Computers and Communications, 2009. ISCC 2009. IEEE Symposium on*, 2009, pp. 361-367.

[40] A. Majeed, L. Ke, and N. Abu-Ghazaleh, "TARP: Timing Analysis Resilient Protocol for Wireless Sensor Networks," in *Wireless and Mobile Computing, Networking and Communications, 2009. WIMOB 2009. IEEE International Conference on*, 2009, pp. 85-90.

[41] B. Alomair, A. Clark, J. Cuellar, and R. Poovendran, "Statistical Framework for Source Anonymity in Sensor Networks," in *IEEE Global Telecommunications Conference. GLOBECOM.*, 2010, pp. 1-6.

[42] C. Honglong and L. Wei, "From nowhere to somewhere: Protecting end-to-end location privacy in wireless sensor networks," in *Performance Computing and Communications Conference (IPCCC), 2010 IEEE 29th International*, 2010, pp. 1-8.

[43] L. Rongxing, L. Xiaodong, Z. Haojin, and S. Xuemin, "TESP2: Timed Efficient Source Privacy Preservation Scheme for Wireless Sensor Networks," in *IEEE International Conference on Communications. ICC.*, 2010, pp. 1-6.

[44] G. Suarez-Tangil, E. Palomar, B. Ramos, and A. Ribagorda, "An experimental comparison of source location privacy methods for power optimization in WSNs," presented at the Proceedings of the 3rd WSEAS international conference on Advances in sensors, signals and materials, Faro, Portugal, 2010.

[45] W. Yang and W. Zhu, "Protecting Source Location Privacy in Wireless Sensor Networks with Data Aggregation," in *Ubiquitous Intelligence and Computing.* vol. 6406, Z. Yu, R. Liscano, G. Chen, D. Zhang, and X. Zhou, Eds., ed: Springer Berlin Heidelberg, 2010, pp. 252-266.

[46] K. Bicakci, H. Gultekin, B. Tavli, and I. E. Bagci, "Maximizing lifetime of event-unobservable wireless sensor networks," *Computer Standards & Interfaces,* 2011, vol. 33, pp. 401-410.

[47] A. Jhumka, M. Leeke, and S. Shrestha, "On the Use of Fake Sources for Source Location Privacy: Trade-Offs Between Energy and Privacy," *The Computer Journal,* June 1, 2011, vol. 54, pp. 860-874.

[48] S. Kokalj-Filipovic, F. Le Fessant, and P. Spasojevic, "The quality of source location protection in globally attacked sensor networks," in *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2011 IEEE International Conference on*, 2011, pp. 44-49.

[49] S. Ortolani, M. Conti, B. Crispo, and R. Di Pietro, "Events privacy in WSNs: A new model and its application," in *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2011 IEEE International Symposium on a*, 2011, pp. 1-9.

[50] Y. Yang, Zhou, J., Deng, R.H., Bao, F., "Better security enforcement in trusted computing enabled heterogeneous wireless sensor networks," *Security and Communication Networks,* 2011, pp. 11-22.

[51] M. M. E. A. Mahmoud and S. Xuemin, "A Cloud-Based Scheme for Protecting Source-Location Privacy against Hotspot-Locating Attack in Wireless Sensor Networks," *Parallel and Distributed Systems, IEEE Transactions on,* 2012, vol. 23, pp. 1805-1818.

[52] L. Kazatzopoulos, C. Delakouridis, G. F. Marias, and P. Georgiadis, "iHIDE: hiding sources of information in WSNs," in *Security, Privacy and Trust in Pervasive and Ubiquitous Computing, 2006. SecPerU 2006. Second International Workshop on*, 2006, pp. 8 pp.-48.

[53] O. Yi, L. Zhengyi, C. Guanling, J. Ford, and F. Makedon, "Entrapping adversaries for source protection in sensor networks," in *World of Wireless, Mobile and*

Multimedia Networks, 2006. WoWMoM 2006. International Symposium on a, 2006, pp. 10 pp.-34.

[54] S. M. a. G. Xue, "Efficient anonymity schemes for clustered wireless sensor networks," *Int. J. Sensor Networks,* 2006, vol. 1.

[55] O. Yi, L. Zhengyi, X. Yurong, N. Triandopoulos, Z. Sheng, J. Ford*, et al.*, "Providing Anonymity in Wireless Sensor Networks," in *Pervasive Services, IEEE International Conference on*, 2007, pp. 145-148.

[56] S. Jang-Ping, J. Jehm-Ruey, and T. Ching, "Anonymous Path Routing in Wireless Sensor Networks," in *IEEE International Conference on Communications. ICC.*, 2008, pp. 2728-2734.

[57] R. Di Pietro and A. Viejo, "Location privacy and resilience in wireless sensor networks querying," *Computer Communications,* 2011, vol. 34, pp. 515-523.

[58] J.-H. Park, Y.-H. Jung, H. Ko, J.-J. Kim, and M.-S. Jun, "A Privacy Technique for Providing Anonymity to Sensor Nodes in a Sensor Network," in *Ubiquitous Computing and Multimedia Applications*. vol. 150, T.-h. Kim, H. Adeli, R. Robles, and M. Balitanas, Eds., ed: Springer Berlin Heidelberg, 2011, pp. 327-335.

[59] S. Min, H. Wenhui, Z. Sencun, C. Guohong, S. Krishnamurth, and T. La Porta, "Cross-layer Enhanced Source Location Privacy in Sensor Networks," in *Sensor, Mesh and Ad Hoc Communications and Networks, 2009. SECON '09. 6th Annual IEEE Communications Society Conference on*, 2009, pp. 1-9.

[60] H. Wang, B. Sheng, and Q. Li, "Privacy-aware routing in sensor networks," *Computer Networks,* 2009, vol. 53, pp. 1512-1529.

[61] F. Yanfei, C. Jiming, L. Xiaodong, and S. Xuemin, "Preventing Traffic Explosion and Achieving Source Unobservability in Multi-Hop Wireless Networks Using Network Coding," in *Global Telecommunications Conference (GLOBECOM 2010), 2010 IEEE*, 2010, pp. 1-5.

[62] F. Yanfei, J. Yixin, Z. Haojin, and S. Xuemin, "An Efficient Privacy-Preserving Scheme against Traffic Analysis Attacks in Network Coding," in *INFOCOM 2009, IEEE*, 2009, pp. 2213-2221.

[63] R. El-Badry, A. Sultan, and M. Youssef, "HyberLoc: Providing Physical Layer Location Privacy in Hybrid Sensor Networks," in *Communications (ICC), 2010 IEEE International Conference on*, 2010, pp. 1-5.

[64] R. El-Badry, M. Youssef, and M. Eltoweissy, "Hidden Anchor: Providing Physical Layer Location Privacy in Hybrid Wireless Sensor Networks," in *New Technologies, Mobility and Security (NTMS), 2009 3rd International Conference on*, 2009, pp. 1-5.

[65] R. Rios and J. Lopez, "Exploiting Context-Awareness to Enhance Source-Location Privacy in Wireless Sensor Networks," *The Computer Journal,* October 1, 2011, vol. 54, pp. 1603-1615.

[66] O. Sangho and M. Gruteser, "Multi-node coordinated jamming for location privacy protection," in *MILITARY COMMUNICATIONS CONFERENCE, 2011*, pp. 1243-1249.

[67] B. Tavli, M. M. Ozciloglu, and K. Bicakci, "Mitigation of Compromising Privacy by Transmission Range Control in Wireless Sensor Networks," *Communications Letters, IEEE,* 2010, vol. 14, pp. 1104-1106.

[68] J. Chen, X. Du, and B. Fang, "An efficient anonymous communication protocol for wireless sensor networks," *Wireless Communications and Mobile Computing,* 2012, vol. 12, pp. 1302-1312.

[69] S. Misra and X. Guoliang, "SAS: A Simple Anonymity Scheme for Clustered Wireless Sensor Networks," in *Communications, 2006. ICC '06. IEEE International Conference on*, 2006, pp. 3414-3419.

[70] B. Alomair, A. Clark, J. Cuellar, and R. Poovendran, "Toward a Statistical Framework for Source Anonymity in Sensor Networks," *IEEE Transactions on Mobile Computing.,* 2013, vol. 12, pp. 248-260.

[71] J. Kong and X. Hong, "ANODR: anonymous on demand routing with untraceable routes for mobile ad-hoc networks," presented at the Proceedings of the 4th ACM international symposium on Mobile ad hoc networking & computing., Annapolis, Maryland, USA, 2003.

[72] Kerckhoffs' Principle. Available from: http://en.citizendium.org/wiki/Kerckhoffs%27_Principle (accessed on 1 March 2015).

[73] Z. Yanchao, L. Wei, L. Wenjing, and F. Yuguang, "Location-based compromise-tolerant security mechanisms for wireless sensor networks," *Selected Areas in Communications, IEEE Journal on,* 2006, vol. 24, pp. 247-260.

[74] L. Rongxing, L. Xiaodong, Z. Chenxi, Z. Haojin, H. Pin-Han, and S. Xuemin, "AICN: An Efficient Algorithm to Identify Compromised Nodes in Wireless Sensor Network," in *IEEE International Conference on Communications. ICC.*, 2008, pp. 1499-1504.

[75] H. Song, L. Xie, S. Zhu, and G. Cao, "Sensor node compromise detection: the location perspective," presented at the Proceedings of the 2007 international conference on Wireless communications and mobile computing, Honolulu, Hawaii, USA, 2007.

[76] L. Tao, S. Min, and M. Alam, "Compromised Sensor Nodes Detection: A Quantitative Approach," in *28th International Conference on Distributed Computing Systems Workshops. ICDCS.* , 2008, pp. 352-357.

[77] S. Zhu, S. Setia, and S. Jajodia, "LEAP+: Efficient security mechanisms for large-scale distributed sensor networks," *ACM Trans. Sen. Netw.*, 2006, vol. 2, pp. 500-528.

[78] I. Mabrouki and A. Belghith, "E-SeRLoc: An enhanced serloc localization algorithm with reduced computational complexity," in *9th International Wireless Communications and Mobile Computing Conference. IWCMC.*, 2013, pp. 153-158.

[79] L. Lazos and R. Poovendran, "SeRLoc: secure range-independent localization for wireless sensor networks," presented at the Proceedings of the 3rd ACM workshop on Wireless security, Philadelphia, PA, USA, 2004.

[80] C. Juan, Z. Hongli, F. Binxing, D. Xiaojiang, Y. Lihua, and Y. Xiangzhan, "Towards Efficient Anonymous Communications in Sensor Networks," in *IEEE Global Telecommunications Conference. GLOBECOM.*, 2011, pp. 1-5.

[81] W. Zheng, S. Gao, L. Qiu, and W. Zhang, "A CDS-based topology control algorithm in energy efficient Clustering," in *31st Chinese Control Conference. CCC.* , 2012, pp. 6605-6610.

[82] D. Hongwei, W. Weili, Y. Qiang, L. Deying, L. Wonjun, and X. Xuepeng, "CDS-Based Virtual Backbone Construction with Guaranteed Routing Cost in Wireless Sensor Networks," *IEEE Transactions on Parallel and Distributed Systems.,* 2013, vol. 24, pp. 652-661.

[83] A. Abduvaliyev, A. S. K. Pathan, Z. Jianying, R. Roman, and W. Wai-Choong, "On the Vital Areas of Intrusion Detection Systems in Wireless Sensor Networks," *IEEE Communications Surveys & Tutorials.,* 2013, vol. 15, pp. 1223-1237.

[84] L. YangXia and G. Ye, "A Survey on Intrusion Detection of Wireless Sensor Network," in *2nd International Conference on Information Science and Engineering. ICISE.*, 2010, pp. 1798-1802.

[85] P. Kamat, Y. Zhang, W. Trappe, and C. Ozturk, "Enhancing Source-Location Privacy in Sensor Network Routing," in *Distributed Computing Systems. ICDCS. Proceedings. 25th IEEE International Conference on*, 2005, pp. 599-608.

[86] P.-C. Wu, "Multiplicative, congruential random-number generators with multiplier " *ACM Trans. Math. Softw.,* 1997, vol. 23, pp. 255-265.

[87] L.-Y. Deng, C. Rousseau, and Y. Yuan, "Generalized Lehmer-Tausworthe random number generators," presented at the Proceedings of the 30th annual Southeast regional conference, Raleigh, North Carolina, 1992.

[88] W. H. Payne, J. R. Rabung, and T. P. Bogyo, "Coding the Lehmer pseudo-random number generator," *Commun. ACM,* 1969, vol. 12, pp. 85-86.

[89] J. Chen, X. Du, and B. Fang, "An efficient anonymous communication protocol for wireless sensor networks," *Wirel. Commun. Mob. Comput.,* 2012, vol. 12, pp. 1302-1312.

[90] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," in *Proceedings of the 33rd Annual Hawaii International Conference on System Sciences.*, 2000.

[91] C. Shuguang, A. J. Goldsmith, and A. Bahai, "Energy-constrained modulation optimization," *Wireless Communications, IEEE Transactions on,* 2005, vol. 4, pp. 2349-2360.

[92] W. Stallings, *Network Security Essentials*, 3rd ed.: Prentice Hall, 2007.